Министерство образования и науки Российской Федерации
Федеральное агенство по образованию
Государственное образовательное учреждение
Высшего профессионального образования
Алтайский государственный технический университет
им. И.И.Ползунова

НАУКА И МОЛОДЕЖЬ – 2009

VI Всероссийская научно-техническая конференция студентов, аспирантов и молодых ученых

СЕКЦИЯ

ИНФОРМАЦИОННЫЕ И ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

подсекция

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И АВТОМАТИЗИРОВАННЫХ СИСТЕМ

Барнаул – 2009

ББК 784.584 (2 Рос 537) 638.1

VI Всероссийская научно-техническая конференция студентов, аспирантов и молодых ученых "Наука и молодежь – 2009". Секция «Информационные и образовательные технологии». Подсекция «Программное обеспечение вычислительной техники и автоматизированных систем». / Алт. гос. техн. ун-т им. И.И.Ползунова. – Барнаул: изд-во АлтГТУ, 2009. – 113 с.

В сборнике представлены работы научно-технической конференции студентов, аспирантов и молодых ученых, проходившей в 24 апреля 2009 г.

Организационный комитет конференции:
Максименко А.А., проректор по НИР – председатель, Марков А.М., зам. проректора по НИР – зам. председателя, Стопорева Т.А. – ответственный секретарь Центра НИРС – секретарь оргкомитета, Кантор С.А., заведующий кафедрой «Прикладная математика» АлтГТУ – руководитель секции.

Научный руководитель подсекции: к.ф.-м.н., профессор, Кантор С.А.
Секретарь подсекции: к.т.н., доцент, Сорокин А.В.

Компьютерная верстка: Сорокин А.В.

СОДЕРЖАНИЕ

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ УПРАВЛЕНИЯ ДВИЖЕНИЕМ КОЛЕСНЫХ МАШИН

Анкудинов И.Ф., Диль А.А. – студенты, Поддубный В.И. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

В наше время бурного развития высоких технологий активно происходит автоматизация большинства производственных процессов. За счет автоматизации увеличивается эффективность производств и уменьшается себестоимость продукции. Сельское хозяйство — одна из важнейших отраслей народного хозяйства, но в то же время одна из самых отсталых в плане использования передовых технологий.

При обработке посевных площадей широко применяются колесные машины: для вспахивания земли, для посева культур, для обработки их удобрениями и химикатами, для полива, для сбора урожая и т. д. Однако, при ручном управлении колесными машинами возникают следующие проблемы:

1. Перекрываемость рядов при параллельном проходе рядов.
2. Необработка некоторых участков поля при неточном выборе траектории движения трактора.
3. Повышенный расход топлива и времени из-за неоптимальной траектории движения и необходимости обрабатывать пропущенные участки поля.
4. Повышенные психомоторные затраты водителя на управление.
5. Невозможность полностью использовать максимально возможную скорость движения вследствие снижения показателей качества движения.
6. Возможность работы только при хорошей видимости.

Исследования показывают, что, благодаря повышению точности выполнения работ, затраты рабочего времени сокращаются в среднем на 7 %, уменьшаются расходы горючего и рационально используются минеральные удобрения и средства защиты растений. Также благодаря данным приборам необработанных гербицидами полос на полях нет.

В рамках решения поставленной задачи был написан программный комплекс.

Программный комплекс состоит из двух программ:

1. Программа, моделирующая управление движением колесных машин. В программе есть возможность задавать траекторию движения машины. На экране схематично изображается машина, движущаяся по «полю» (вид сверху), программа анализирует текущее положение машины (направление движения и угол поворота колес) и принимает решение о повороте руля на угол, необходимый для поддержания движения по заданной траектории. Задаваемая и реальная траектории движения визуально отображаются на экране.

В программе регулируются основные параметры системы:

- Габариты машины.
- Максимальный угол поворота колес.
- Параметры регулирования.
- Скорость движения машины при моделировании.

2. Программа для управления движением реальной колесной машины, состоящая из следующих модулей:

– Модуль позиционирования. В реальном времени определяет координаты машины на основании данных, получаемых от системы глобального позиционирования — GPS.

– Модуль взаимодействия с аналоговыми приборами. В реальном времени получает аналоговый сигнал (напряжение) с датчика угла поворота колес и преобразовывает его в цифровой вид. Также преобразовывает управляющий сигнал из цифрового в аналоговый и выдает напряжение на электроусилитель руля для поворота колес на необходимый угол.

– Модуль управления. Анализирует координаты, полученные от модуля позиционирования, и текущий угол поворота колес и рассчитывает управление для движения по заданному маршруту.

ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ КОМПОНЕНТОВ ИНСТРУМЕНТАЛЬНОЙ СРЕДЫ ДЛЯ МОДЕЛИРОВАНИЯ И ИССЛЕДОВАНИЯ ПРОЦЕССОВ СБОРА И ОБРАБОТКИ ДАННЫХ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ

Бабенко А.Н., Дубинина Е.В. – студенты, Бочкарева Е.В. – аспирант
Сучкова Л.И. – к.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

На сегодняшний день на рынке программного обеспечения в области обслуживания SCADA-систем существует множество проектов, предназначенных для сбора информации в сложных многоуровневых распределенных системах. К сожалению, при проектировании новых SCADA или при модернизации старых разработчики не застрахованы от ошибок, несмотря на тщательные расчеты и тестирование систем специалистами. Такие ошибки влекут за собой порой огромные материальные расходы, связанные с перенастройкой устройств и, возможно, даже с отказом от разработанной архитектуры системы. В таких случаях очевидным решением является имитационное моделирование ситуаций, происходящих в SCADA-системах. В связи с этим возникает интерес создания программы, обеспечивающей быстрое проектирование распределенных систем с последующей возможностью проверки их работы, не требующих материальных затрат.

Нами был разработан программный продукт, предназначенный для конструирования и моделирования работы распределенных систем. Программа позволяет с помощью графического пользовательского интерфейса спроектировать систему произвольной сложности и архитектуры, элементы которой могут являться различными устройствами трех основных типов: технологический компьютер (ТК), программируемый микроконтроллер (PLC) или датчик. Устройства могут быть связаны каналами. Для каждого из объектов системы (устройств и каналов) есть возможность задания их конфигураций. Конфигурирование устройств и каналов также осуществляется посредством графического пользовательского интерфейса.

Одной из основных функциональных возможностей программы является то, что пользователь может задавать логику работы устройств, то есть определять процессы, выполняемые на устройствах. Процессы используют алгоритмы и функции, которые также задаются пользователем (или могут быть выбраны из списка стандартных). С помощью алгоритмов и функций задаются такие операции, как генерация данных (моделирование процесса измерения датчиками некоторых параметров, например, температуры), формирование пакетов данных, получение данных от других источников (устройств или процессов расчета других величин на том же устройстве), фильтрация, архивирование, восстановление, анализ и отправка данных. Каждый процесс в моделируемой системе логически привязан к устройству, на котором он «исполняется».

Имитация работы спроектированной системы осуществляется с учетом ведения внутреннего модельного времени. С помощью настройки значения условной временной единицы пользователь может регулировать абсолютную скорость моделирования. При имитации работы устройств учитывается их производительность, объемы энергозависимой и энергонезависимой памяти. При имитации передачи данных по каналам связи между устройствами учитывается их пропускная способность. Пользователь программы имеет возможность приостанавливать работу отдельных устройств и каналов связи, тем самым имитируются ситуации поломки и разрывов соединений.

Все события, происходящие в системе, регистрируются. Сведения о состоянии устройств, каналов, пакетах и данных хранятся для каждой сессии моделирования и доступны пользователю в виде логов системы и графиков изменения данных во времени. Таким образом, пользователь может отследить движение пакетов и данных в системе, а также, что самое важно для использования моделирования, может выявить узкие места спроектированной системы, причины возникновения сбоев в логике работы, утечек данных.

Программа имеет возможность создавать новые проекты, а также сохранять и изменять уже спроектированные системы.

Список литературы

1. Андреев, Е.Б., Куцевич, Н.А. SCADA-системы: взгляд изнутри. –М.: РТСофт, 2004 – 176с.
2. Анашкин, А.С., Кадыров, Э.Д., Хазаров, В.Г. Техническое и программное обеспечение распределенных систем управления / Под ред. проф. В.Г. Хазарова – СПб.: П-2, 2004 – 368с.

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РАСЧЕТА ТЕХНОЛОГИЧЕСКИХ ПАРАМЕТРОВ ПРИ ПРОИЗВОДСТВЕ СВАРНЫХ КОНСТРУКЦИЙ

Барбарош А.Е. – студент, Кантор С.А. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В настоящее время и по прогнозам на ближайшие десятилетия основными технологическими вариантами сварки являются и будут являться способы дуговой сварки. Из них наиболее перспективными считаются способы сварки в углекислом газе и под слоем флюса. В связи с этим актуальной является проблема качественного изучения студентами указанных способов сварки.

На сегодняшний день практически все существующие виды сварочных процессов, основаны на применении технологии концентрированного нагрева участков изделий до температуры плавления или до пластического состояния. Нарушение технологии (например, неправильный нагрев или охлаждение) может явиться причиной появления различных дефектов (трещины, недовары и др.). Однако если не учитывать результаты анализа теплового состояния металла, нельзя достаточно достоверно объяснить большую часть наблюдаемых явлений.

Разрабатываемая в приложении программа предназначена в помощь студентам, обучающимся сварочному производству, в изучении свойств сварки в среде углекислого газа и под слоем флюса, а так же в изучении свойств тепловых процессов, протекающих при производстве сварных конструкций.

Первая часть программы предназначена для расчета параметров тепловых процессов. Она позволяет производить расчеты для трех математических моделей.

Первая модель предполагает, что сварка производится на полубесконечном теле (то есть ограниченном только по оси Oz в одном направлении), все физические размеры источника пренебрежимо малы (точечный источник), его действие на тело непрерывно, а скорость движения не очень высока (как правило, не превышает 1см/с).

Для следующих моделей считаем, что линейный (два параметра пренебрежимо малы по сравнению с третьим) источник непрерывно действует на пластину (тело, ограниченное только по оси Oz). При этом для второй модели источник сварки считаем подвижным, а для третьей быстродвижущимся.

Формула мгновенной скорости охлаждения W имеет вид:

$$W = W(\lambda, c_\gamma, \nu, T_0, \eta, I, U, \delta), \quad (1)$$

где

λ – коэффициент теплопроводности, Вт/см*К ;

c_γ – коэффициент теплоемкости, Дж/см³*К;

ν – скорость сварки, см/с;

T_0 – начальная температура, С

η – эффективный коэффициент полезного действия, %

I – сила тока, А

U – напряжение дуги, В,

δ – толщина пластины, мм.

При анализе холодных трещин необходимо, используя формулу (1), рассчитать действительную мгновенную скорость охлаждения, сравнить ее с критическим значением, сделать вывод о возникновении холодных трещин сварки, и в случае их возникновения пересчитать начальную температуру.

Для расчета температуры используем формулу вида

$$T = T(x, y, \lambda, c_\gamma, \nu, T_0, \eta, I, U, \delta, \alpha, a). \quad (2)$$

Здесь

T_0 – начальная температура, С

α – коэффициент теплоотдачи, см²/с;

a – коэффициент температуропроводности, см²/с;

x и y – координаты требуемой точки относительно центра.

Для того, чтобы выполнить анализ температур, необходимо построить графики зависимости температуры T от координат x,y, а так же времени t (в этом случае одна координата становится функцией времени) в различных разрезах (например, температуры точек, лежащих на оси позади движущейся дуги).

При работе с термическими циклами точек необходимо, приняв в формуле (2) $x = V*t$, построить график зависимости $T(t)$ и наложить его на графики, полученные экспериментально, после чего выполнить их сравнение.

Уравнение зависимости температуры от глубины проплавления z имеет вид

$$T = T(T_0, \eta, I, U, \lambda, \nu, a, c_\gamma, \delta, \alpha, z). \quad (3)$$

Для того чтобы рассчитать глубину и ширину проплавления, необходимо решить это уравнение относительно z.

При построении поверхностных изотерм необходимо вначале определить коэффициент распределения. Далее, используя номограмму построения температурных полей предельных состояний, определить значения радиус-векторов при различных углах от точки расположения источника тепла. По этим значениям строится искомая изотерма.

Пример работы первой части программы представлен на рисунке 1.

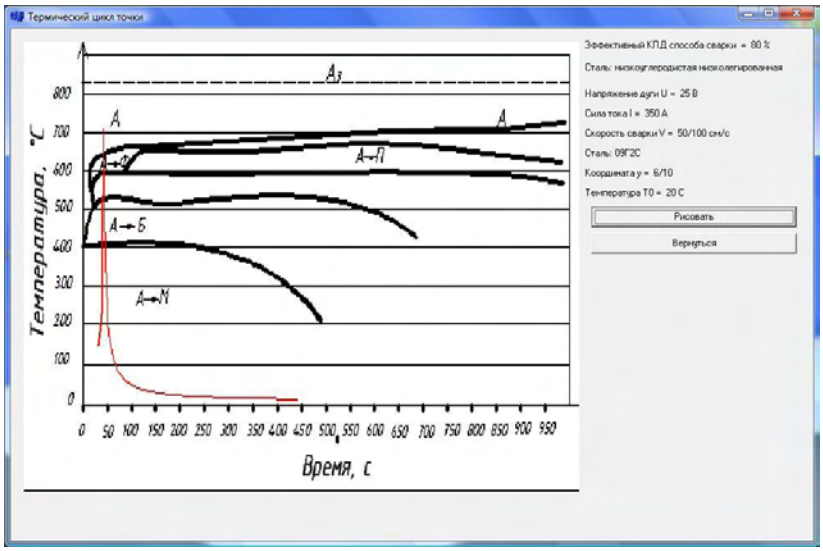


Рисунок 1 – пример работы первого модуля программы

Вторая часть программы предназначена для расчета технологических параметров при применении сварки в среде углекислого газа и под слоем флюса.

В качестве исходных величин для проведения расчетов были выбраны тип сварного соединения, марка сварочной проволоки (по ней определяются физические характеристики проволоки), толщина свариваемых элементов, диаметр сварочной проволоки, величина зазора между свариваемыми элементами, а так же ряд других параметров, изменение которых может представлять определенные трудности.

На основании этих величин определяются значения целого ряда выходных параметров. Некоторые из них, такие как расход углекислоты и максимальная величина сварочного тока, определяются по таблицам. Другие, например напряжение дуги, скорость сварки или коэффициент угара и разбрызгивания, рассчитываются.

Пример работы второй части программы представлен на рисунке 2.

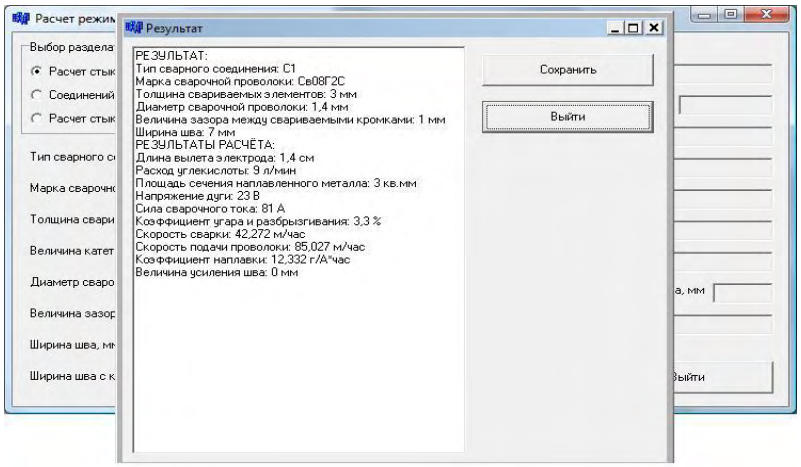


Рисунок 2 – пример работы второй части программы

РАЗРАБОТКА ПОИСКОВОЙ СИСТЕМЫ ДЛЯ WEB-РЕСУРСА АКУНБ ИМ. В.Я. ШИШКОВА

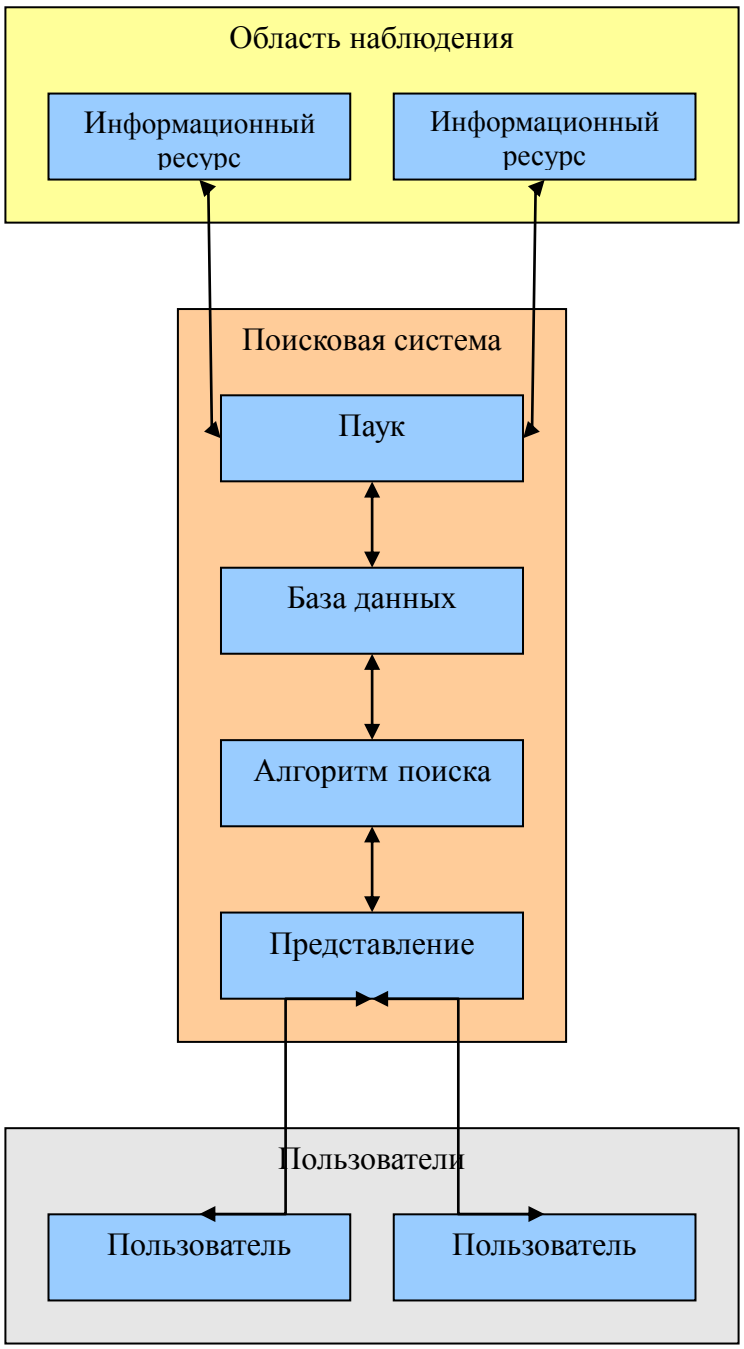
Беккер А.В. – студент, Лукоянычев В.Г. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Поисковые механизмы подразделяют на 2 основных типа:

1. проводящих поиск путем просмотра всех документов области наблюдения при каждой операции поиска. Такие механизмы применимы имеют существенное ограничение - область наблюдения должна быть маленькой, причем размер области пропорционален скорости доступа к ней. В связи с тем, что скорости в интернете в пересчете на потребителя достаточно малы и трафик имеет свою стоимость, использование данного типа невозможно для целей работы невозможно.
2. индексирующие поисковые механизмы - сохраняют всю интересующую информацию из области наблюдения в свою базу данных, которая потом используется для проведения поиска.

Таким образом, целью работы будет создание индексирующей поисковой системы, применимой для использования как на отдельно взятом сайте, так и на некоторой группе сайтов.

Абстрактно структуру можно представить следующим образом:



Область наблюдения - множество всех информационных источников, среди которых необходимо проводить поиск. Информационными источниками могут выступать web-сайты.

Паук - некий механизм, который по указанным критериям опрашивает информационные источники и сохраняет/обновляет всю необходимую для поиска информацию в базе данных. В качестве критерия могут выступать таймер, принудительный запуск пользователем и др.

База данных - база данных, используемая для хранения всей необходимой для поиска информации. В её качестве может выступать практически любая современная база данных с хорошим масштабированием.

Алгоритм поиска - основная интеллектуальная составляющая системы. Именно здесь реализуются выбор подходящих под критерии поиска элементов информационных ресурсов. Источником данных для него является база данных.

Представление - механизм запроса и представления информации пользователю. Обычно - web-интерфейс. Преобразует информацию между внутренним и внешним видом, например, выводит результаты в виде нескольких html-страниц.

Пользователь - сущность, которая обращается к поисковой системе. Обычно подразумевается, что это браузер, которым управляет человек, хотя может быть произвольной системой, использующая протокол взаимодействия с представлением.

Основной особенностью классических поисковых систем является то, что они проводят поиск только текстовой информации. В связи с этим, паук разбивается на две составляющие:

1. парсер - преобразует элемент информационного ресурса к текстовому виду. В роли элемента может выступать html-страница, документ word, pdf и др.
2. анализатор - выделяет всю полезную информацию из выходной информации парсера и добавляет её в базу данных.

Таким образом, для распознавания системой нового типа ресурсов достаточно реализовать лишь новый парсер. На начальном этапе разработки предполагается использование только одного парсера - web-страниц.

Анализатор на первых этапах будет просто разбивать входную информацию на слова и заносить её в базу данных с отметкой в каком порядке и на каком элементе какого ресурса был найден.

ЗАДАЧИ АВТОМАТИЗАЦИИ КОНТРОЛЯ ПАРАМЕТРОВ ПРИ ИССЛЕДОВАНИИ ПАЦИЕНТОВ НА НАЛИЧИЕ ОНКОЛОГИЧЕСКИХ ЗАБОЛЕВАНИЙ

Белоножка П. А. – аспирант, Астахова А. В. – к.э.н., профессор
Алтайский государственный технический университет (г. Барнаул)

На сегодняшний день Алтайский край занимает одно из первых мест в России по показателям онкологических заболеваний. В среднем по краю показатель составляет 403,6 на 100 тысяч населения, тогда как по России 333,7 на 100 тысяч населения. Ежегодно эти показатели продолжают расти.

Рост заболеваемости и смертности в этой области является серьезной социально-экономической проблемой. Решение этой проблемы заключается в создании комплексного подхода, включающего в себя меры, направленные на улучшение сложившейся ситуации и повышение эффективности оказания медицинской помощи. Для этого необходимо исследование медико-биологических процессов, способствующих росту заболеваемости, а также детальное изучение методик оказания лечебно-профилактической помощи пациентам.

Анализ текущего положения позволяет говорить о том, что изменение количественных и качественных характеристик заболеваемости в лучшую сторону возможно в основном благодаря выявлению заболеваний на ранней стадии, а также проведению различных профилактических мероприятий. В этой связи имеет значение определение группы риска для каждого пациента.

Постановка диагноза и выбор методик лечения, основанные на комплексных знаниях и опыте врача-эксперта, являются важнейшим этапом при оказании помощи больным. В следствии этого создание модели принятия решений по каждому пациенту индивидуально имеет существенную практическую значимость.

Особое внимание последние годы уделяется методам оценки эффективности деятельности онкологической службы. Основными критериями оценки эффективности борьбы с раковыми заболеваниями являются выживаемость больных, заболеваемость и смертность, распространенность, запущенность, активная выявляемость и так далее. Все эти показатели напрямую зависят от качества учета анамнеза онкологических больных.

В связи с этим, важную роль приобретают не только автоматизация процесса

регистрации и хранения информации, но и ее грамотная обработка, статистический анализ. Чтобы не сделать ложных выводов, следует оценивать также адекватность результатов компьютерных исследований.

Показатели, применяемые в онкологии, условно можно разделить на «показатели заболеваемости и смертности», «показатели выживаемости» и «показатели состояния онкологической помощи». Одной из наиболее важных характеристик эффективности деятельности онкологических служб являются показатели первой группы.

Остановимся на расчете показателей заболеваемости и смертности.

Для выражения частоты заболевания (смерти) среди населения используется несколько показателей, которые рассчитываются на 100000 населения соответствующего пола и возраста.

«Грубый» (все возрасты) и возрастной показатели

Такой показатель (C) на 100000 населения рассчитывается делением общего числа случаев (R) на численность населения (N) и умножается на 100000.

$C = R / N * 100000$

Возрастной показатель

Такой показатель (a_i) для i-й возрастной группы рассчитывается делением числа случаев в возрастной группе (r_i) на соответствующую численность населения (n_i) и умножением результата на 100000.

$a_i = r_i / n_i * 100000$

Стандартизация по возрасту

Итоговый показатель стандартизации по возрасту позволяет оценивать заболеваемость конкретным злокачественным новообразованием в нескольких группах населения с разным возрастным составом или в одной возрастной группе с течением времени. Для этого применяется метод прямой стандартизации. Стандартизованный по возрасту показатель — это теоретический показатель, который может быть получен при использовании наблюдаемых возрастных показателей среди специальной группы населения, называемой стандартным населением. Чаще всего применяется мировой стандарт возрастного распределения, рекомендуемый Международным агентством по изучению рака.

Таким образом, стандартизованный по возрасту показатель (STAND) рассчитывается как сумма произведений a_i*w_i по i от 1 до A, деленная на сумму w_i по i от 1 до A, где w_i - численность возрастной группы i стандартного населения, а a_i — показатель соответствующей возрастной группы.

$STAND = (\Sigma a_i*w_i) / (\Sigma w_i)$

Кумулятивный показатель и кумулятивный риск

Кумулятивный показатель представляет собой сумму возрастных показателей заболеваемости за каждый год (например, от рождения до определенного возраста).

Рассчитывается такой показатель с помощью следующей приближенной формулы:

Кум. показатель = Σa_i*t_i, где a_i — возрастной показатель заболеваемости в i-й возрастной группе (от 1 до A), насчитывающей t_i лет. Сумма рассчитывается до возрастной группы A.

Кумулятивный риск является риском развития конкретного злокачественного новообразования, которому лицо подверглось бы в течении определенного периода жизни, при условии отсутствия всех прочих причин смерти. Период жизни, за который аккумулируется риск представляет обычно интервал от 0 до 74 лет для взрослых и от 0 до 14 лет для детей. Величина выражается в процентах.

Кум. риск = 100 * [1 — exp(-кумулятивный показатель/100)]

Вариационные ряды

Вариационный ряд — это статистическая совокупность, показывающая распределение

изучаемого явления по величине какого-либо количественного признака (например, распределение числа заболеваний по возрасту). В таких рядах количественно изменяемый признак носит название варьирующего (возрастные группы), а отдельные его количественные выражения называются вариантами (случай заболевания). Числа, показывающие частоту, с которой встречаются варианты в составе данной совокупности, носят называние частот (число заболеваний). Вся совокупность может быть разбита на группы единиц (частоты), имеющие одинаковые размеры вариант. Ряд, в котором сопоставлены варианты и соответствующие им частоты, то есть отражающий распределение изучаемой совокупности по величине варьирующего признака, носит название вариационного ряда [1].

Вариационный ряд характеризуется средними величинами (средняя арифметичская, медиана и тому подобные).

Средняя арифметическая

Рассмотрим расчет средней арифметической на примере среднего возраста заболевших. Для того, чтобы выполнить его следует перемножить значения частот и соответствующих им вариант, полученные произведения суммировать и разделить на общую сумму частот.

Средняя арифметическая = $(\Sigma v_i * r_i) / (\Sigma r_i)$

Медиана

Медиана представляет собой варианту, делящую вариационный ряд на две равные половины. Медиана вычисляется при помощи так называемого начетного ряда.

Динамические ряды

Динамический ряд — это совокупность однородных статистических величин, показывающих изменение какого-либо явления на протяжении определенного промежутка времени. При анализе динамического ряда следует различать несколько характеризующих его показателей. Прежде всего, ряд может быть охарактеризован самими величинами членов ряда, так называемым абсолютным уровнем.

Скорость ряда, то есть величина прироста (или уменьшения) уровня ряда в единицу времени характеризует динамику ряда в большей степени, чем абсолютный уровень. Скорость ряда получается путем вычитания последующего члена ряда из предыдущего. Если динамический ряд возрастает, показатель скорости имеет положительное значение. В убывающем ряду он выражается отрицательным числом.

Следует также отдельно учитывать стандартную ошибку (s.e.) при расчете показателей заболеваемости и смертности. Для каждого метода s.e. имеется свой способ, позволяющий оценить достоверность результатов.

Расчет показателей заболеваемости и смертности с помощью подобных методик позволяет обеспечивать сопоставимость результатов анализа онкологической информации с международными данными, а также может быть использован для оценки эффективности работы онкологических служб и разработки противораковых мероприятий.

Приведенные выше методические рекомендации с точки зрения автора данной статьи не являются достаточными для решения задачи мониторинга пациента на ранней стадии заболевания, а также моделирования и прогнозирования возможных исходов заболевания. Автором данной работы в настоящее время начато исследование в вышеозначенном направлении.

Список литературы

1. Организация онкологической службы в России (методические рекомендации, пособия для врачей) Часть 2 / под редакцией В.И.Чиссова, В.В.Старинского, Б.Н.Ковалева — М.:ФГУ МНИОИ им. П.А.Герцена Росмедтехнологии. - 2007

РАСЧЕТ СРЕДНИХ ЗНАЧЕНИЙ И ВАРИАЦИЙ ОДНОРОДНЫХ ДАННЫХ НА ОСНОВЕ СПЕКТРАЛЬНОЙ МАТРИЧНОЙ НОРМЫ

Бобров А.В. – аспирант
Алтайский государственный технический университет (г. Барнаул)

В докладе обсуждаются вопросы анализа однородных данных, представленных в виде матрицы A размеров $n \times m$ с элементами a_{ij}, принадлежащими заданному интервалу допустимых значений $\Delta = [a_{\min} \quad a_{\max}]$.

Обычно вычисляют взвешенное по строкам и столбцам среднее значение c и вариацию данных d в матрице A

$$c = \sum_{i=1}^{n}\sum_{j=1}^{m} v_i w_j a_{ij}, \qquad d = \left(\frac{1}{nm} \sum_{i=1}^{n}\sum_{j=1}^{m} (c - nm v_i w_j a_{ij})^2 \right)^{1/2},$$

где v_i, w_j – весовые коэффициенты,

$$\sum_{i=1}^{n} v_i = 1, \qquad v_i \geq 0, \qquad i = 1,...,n,$$

$$\sum_{j=1}^{m} w_j = 1, \qquad w_j \geq 0, \qquad j = 1,...,m.$$

Заметим, что значение c является решением задачи оптимизации

$$c = \arg\min_{s \in \Delta} \left\| B(s) \right\|_F,$$

где

$$\left\| B(s) \right\|_F = \left(\sum_{i=1}^{n}\sum_{j=1}^{m} (b_{ij}(s))^2 \right)^{1/2}$$

норма Фробениуса матрицы $B(s)$ с элементами $b_{ij}(s) = s - nm v_i w_j a_{ij}$. При этом

$$d = \frac{1}{\sqrt{nm}} \left\| B(c) \right\|_F.$$

В вычислительной математике, методах обработки данных [1,2] наряду с нормой Фробениуса применяются и другие матричные нормы, например, гельдеровы нормы:

$$\left\| B \right\|_1 = \max_j \sum_{i=1}^{n} \left| b_{ij} \right|, \qquad \left\| B \right\|_2 = \sigma_{\max}(B), \qquad \left\| B \right\|_\infty = \max_i \sum_{j=1}^{m} \left| b_{ij} \right|,$$

где $\sigma_{\max}(B)$ – максимальное сингулярное число матрицы B. В расчетах на реальных данных часто применяется 2-норма (спектральная норма), поскольку данная норма обладает свойством грубости (робастности) по отношению к изменениям в элементах матрицы [2]. Изменения (возмущения) в матрице данных возможны по ряду причин: технические ошибки при записи данных, отсутствие данных и др.

Сингулярные числа матрицы A размеров $n \times m$, упорядоченные по невозрастанию, обозначим $\sigma_1(A) \geq \sigma_2(A) \geq ... \geq \sigma_{\min(n,m)}(A)$. Справедливы следующие утверждения [2].

Утверждение 1. Пусть матрицы P, Q, R размеров $n \times m$ связаны соотношением $P = Q + R$. Тогда

$$\left| \sigma_i(P) - \sigma_i(Q) \right| \leq \sigma_1(R), \qquad i = 1,...,\min(n,m).$$

Утверждение 2. Пусть матрица Q получена из матрицы P вычеркиванием строки или столбца. Тогда $\sigma_1(P) \geq \sigma_1(Q) \geq \sigma_2(P)$.

Рассмотрим два вида возмущений в матрице данных: ошибки в записи элементов матрицы и пропуски строк или столбцов. Из утверждений 1 и 2 следуют:

Утверждение 3. Пусть две матрицы данных A и $\overline{A}$ отличаются k элементами, т.е.

$$\overline{A} = A + \sum_{i=1}^{k} A_i \delta_i \ ,$$

где в матрицах A_i все элементы равны нулю, за исключением одного, равного единице. Составим вектор возмущений $\delta = [\delta_1, \ \ldots, \ \delta_k]$. Тогда для любого s справедлива оценка

$$\left| \left\| \overline{B}(s) \right\|_2 - \left\| B(s) \right\|_2 \right| \le \left\| \delta \right\|_2 ,$$

где $\left\| \delta \right\|_2 = \left(\sum_{i=1}^{k} \delta_i{}^2 \right)^{1/2}$.

Утверждение 4. Пусть в результате удаления строки или столбца в матрице данных A получена матрица $\overline{A}$. Тогда для любого s справедлива оценка

$$\left\| B(s) \right\|_2 = \sigma_1(B(s)) \ge \left\| \overline{B}(s) \right\|_2 \ge \sigma_2(B(s)) .$$

Мы предлагаем для оценки среднего значения и вариации данных в двумерном массиве использовать спектральную норму. Среднее значение определим как решение задачи оптимизации

$$c = \arg\min_{s \in \Delta} \left\| B(s) \right\|_2 .$$

Вариацию данных будем вычислять по формуле

$$d = \frac{1}{\sqrt{nm}} \left\| B(c) \right\|_2 .$$

Определить значения c и d достаточно просто с использованием систем компьютерной математики, например, MATLAB или SCILAB. В докладе обсуждаются результаты тестовых расчетов, проведенных на данных модульно-рейтинговой системы квалиметрии учебной деятельности студентов АлтГТУ. Проводится сравнение значений c и d, полученных на основе нормы Фробениуса и спектральной нормы. Рассматривается зависимость оценок средних значений и вариаций рейтингов студенческих групп от ошибок в записи данных.

Список литературы

1. Воеводин В.В. Кузнецов Ю.А. Матрицы и вычисления. – М.: Наука, 1984.
2. Лоусон Ч., Хенсон Р. Численное решение задач метода наименьших квадратов. – М.: Наука, 1986.

РАЗРАБОТКА ПРОГРАММЫ РАСЧЕТА КОНТАКТНЫХ АППАРАТОВ ДЛЯ КАТАЛИТИЧЕСКОГО ОКИСЛЕНИЯ ДИОКСИДА СЕРЫ

Богомолова И.В. – студентка, Свит Т.Ф. – к.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Серная кислота является одним из основных продуктов химической промышленности и применяется в различных отраслях народного хозяйства. Исходным веществом для производства серной кислоты является оксид серы SO_3, который образуется в результате сжигания серы или другого серосодержащего сырья. В производстве серной кислоты контактным методом получение SO_3 окислением SO_2 происходит по реакции

$$SO_2 + ½ O_2 <=> SO_3 + Q$$

в присутствии катализатора. В промышленных условиях существенное значение имеет скорость окисления SO_2.

Для осуществления процесса окисления SO_2 применяются различные контактные аппараты. В последнее время преимущественное распространение получили контактные аппараты с промежуточным охлаждением, которые отличаются простотой, возможностью использования тепла реакции, малым гидравлическим сопротивлением и другими достоинствами. В этих аппаратах имеются теплообменники, которые размещены внутри контактного аппарата или вне него.

Основными показателями хорошей работы контактного отделения являются высокая степень превращения и некоторый избыток тепла в системе, гарантирующий устойчивость её температурного режима без включения подогревателя даже при небольших остановках системы и понижении концентрации SO_2 в газе. В практических условиях степень превращения зависит главным образом от температурного режима. При точном регулировании температуры на всех участках контактного аппарата, обеспечивается постоянная высокая степень превращения. Необходимо разработать программу, позволяющую рассчитать такой оптимальный режим каждого слоя контактной массы, при котором общее время контакта будет минимальным, а степень превращения максимальной.

Программа предназначена для студентов Алтайского государственного технического университета, а также других технических вузов, обучающихся по специальности «Технология неорганических веществ». Кроме того, данной программой могут пользоваться преподаватели, как для написания заданий студентам, так и для демонстрации студентам результатов расчетов для контактных аппаратов различного типа.

Применение программного продукта позволяет решить следующие задачи:

- Расчет оптимального режима заданного контактного аппарата
- Расчет материального баланса заданного контактного аппарата
- Расчет контактной массы заданного контактного аппарата
- Просмотр результатов расчета
- Формирование HTML-отчета о произведенном расчете

Программа позволяет пользователю самому выбрать интересующий тип контактного аппарата, а так же задать его параметры и начальные параметры вводимого газа; после этого программа рассчитает результаты и представит их в удобном для пользователя виде.

Для программного продукта разработано руководство пользователя, позволяющее в короткие сроки полностью освоить работу с программой.

Система написана и реализована под IBM – совместимые компьютеры с операционной системой Windows 2000/XP/Vista.

Программный продукт реализован на визуальной среде программирования Microsoft Visual Studio 2005 под платформу Microsoft .Net Framework 2.0 и поэтому все новые изменения, исправления, а так же добавление расчета новых параметров может быть произведено за короткий период времени.

Программный продукт обладает удобным графическим интерфейсом под ОС Windows, который обеспечивает наиболее удобную работу с ним. Ввод информации реализован с помощью . Для каждого пункта меню в программе имеется вспомогательная информация. Результаты, полученные в ходе работы программы, можно увидеть на экране или вывести в HTML-файл. Программный продукт имеет интерактивный графический интерфейс и защиту ввода от ошибок пользователя, что обеспечивает высокую отказоустойчивость системы.

РАЗРАБОТКА ПРОКСИ-СЕРВЕРА ДЛЯ ПРОТОКОЛА HTTP С КЕШИРОВАНИЕМ И АВТОРИЗАЦИЕЙ

Большаков Е.Ф. – студент
Алтайский государственный технический университет (г. Барнаул)

В настоящее время каждый день по сети интернет передаются очень большие объемы данных. Для работы даже с частью этих данных необходим широкополосный доступ к сети интернет, но зачастую скорость передачи данных по нему ограничена одним или двумя мегабитами в секунду. К тому же к этому каналу подключается не один компьютер, а вся компьютерная сеть. Если в сети будет находиться хотя бы десять компьютеров и каждый пользователь попытается получить доступ к интернет ресурсу, то в итоге канал с пропускной способностью в один мегабит в секунду разделится между ними всеми и каждому достанется примерно только по сто килобит в секунду, а это потеря драгоценного времени. При подключении локальной сети к сети интернет встает еще задача защиты от вторжений из вне.

Одним из решений проблемы обмена данными по сети интернет и защитой локальной сети является использование прокси-серверов. В подавляющем большинстве пользователи использует только веб-сервисы, предоставляемые через сеть интернет, а данные передаваемые во время работы с ними можно кэширова и при повторном обращении пользователей к какому-либо ресурсу будет использована его локальная копия, которая расположена уже в локальной сети. Пропускная способность локальной сети гораздо выше пропускной способности канала интернет, к тому же если берется копия ресурса хранящаяся на прокси-сервере, то канал интернет остается свободным для других пользователей.

Прокси-сервер также позволяет вести контроль доступа и учет использования пользователем сети интернет.

Таким образом, прокси-сервер предусматривает решение следующих задач:

- ведение журнала посещаемости сети интернет учениками и учителями,
- ограничение доступа к сети интернет, как отдельным пользователям, так и группам пользователей,
- журналирование ресурсов посещенных пользователями,
- кэширование данных,
- авторизация пользователей.

Прокси-сервер будет работать на трех портах. Один из них будет задействован для передачи данных между прокси-сервером и клиентской программой для авторизации пользователей. Оставшиеся два порта будут использованы для проксирования HTTP протокола, но они будут использовать различные методы авторизации пользователей, один для общего использования будет использовать авторизацию, которая проводится с использованием клиентской программы, а другой для сервисного обслуживания (например, обновление антивирусных баз) будет использовать метод авторизации пользователей, который предлагает протокол HTTP.

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ВЕДЕНИЯ РЕГИСТРА БОЛЬНЫХ С КЛАПАННОЙ БРОНХОБЛОКАЦИЕЙ

Борисова И.В. – студентка, Леонов С.Л. – к.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В последние годы в России, как и во многих странах мира, увеличилось число тяжёлых, быстро прогрессирующих и нередко приводящих к летальному исходу форм туберкулёза лёгких. Одной из причин сложившейся ситуации является нарастание частоты случаев, вызванных лекарственно-устойчивыми штаммами микобактерий туберкулёза. Лечение больных туберкулёзом, выделяющих лекарственно-устойчивые штаммы, затруднено и

недостаточно эффективно, поэтому в этих условиях существенно возрастает роль немедикаментозных методов лечения, в частности, искусственного пневмоторакса. Однако для лечения искусственным пневмотораксом имеется ряд противопоказаний, таких как остро прогрессирующие формы туберкулеза легких, туберкулез бронхов, эмпиема плевры, нарушение свертываемости крови и другие. Кроме того, тяжелым, а нередко смертельным осложнением ряда заболеваний является легочное кровотечение, лечение которого с применением общепринятой гемостатической терапии эффективно лишь при кровотечениях диапедезного характера (кровохарканьях) и малых кровотечениях, тогда как ее эффективность при средних и больших рецидивирующих кровотечениях вызывает серьезные сомнения.

На основании многолетних научных исследований и проведенных клинических испытаний, был разработан метод лечения туберкулёза лёгких и его осложнений путём применения эндобронхиального клапана. Этот метод получил название клапанной бронхоблокации (КББ).

Метод развивается, и по данной тематике проводятся научные исследования. Активное внедрение клапанной бронхоблокации в краевом противотуберкулезном диспансере требует выполнения работ по систематизации данных по пациентам и получения статистической информации. Обработка данных вручную занимает значительное время и отрывает врачей от выполнения их прямых обязанностей. Поэтому необходима автоматизация и создание специального ПО.

Разрабатываемая программа предназначена для хранения и обработки данных по больным, при лечении которых был применен метод КББ. Программа представляет собой базу данных, и была написана – согласно требованию заказчика – краевого противотуберкулезного диспансера - с использованием среды разработки Visual For Pro 9.0.

Основной этапы работы пользователя с программой:

1. Ввод данных о больных с КББ. На данном этапе заполняют:
   1.1. Сведения о пациенте (ФИО, пол, дата рождения, место жительства, номер истории болезни, даты поступления и выписки и т.д.);
   1.2. Сведения о его состоянии на момент поступления (диагноз, жалобы, информация о сопутствующих заболеваниях, результаты обследования);
2. Ведение больных. На этом этапе врач указывает:
   2.1. Сведения о проведенном лечении (кроме информации о применении клапанной бронхоблокации, сюда относится информация о дополнительно перенесенных операциях и применяемых препаратах, а так же динамика клинических и лабораторных показателей после КББ).
   2.2. Данные по мониторингу (результаты обследований пациента после выписки).
3. Ведение справочников. К ним относятся:
   3.1. Список операций. В этом справочнике хранится список возможных операций.
   3.2. Список осложнений. В этом справочнике хранится список возможных послеоперационных осложнений.
   3.3. Список отделений. В этом справочнике хранится список отделений лечебного учреждения.
   3.4. Список препаратов. В этом справочнике хранится список возможных препаратов.
   3.5. Список проявлений туберкулеза. В этом справочнике хранится список возможных внелегочных проявлений туберкулеза
   3.6. Список заболеваний. В этом справочнике хранится список возможных заболеваний.
   3.7. Список временных промежутков. В этом справочнике хранится список временных промежутков, через которые может быть проведено обследование.

В основе всех вкладок и таблиц лежат документы, используемые врачами (истории болезни и другая медицинская документация). Это сделано для того, чтобы пользователи, не имеющие большого опыта работы с подобными продуктами, не испытывали затруднений при

работе с ней.

Следующим этапом разработки программы должно стать добавление различных отчетов, которые необходимы как для ведения статистики по больным, так и для научной работы на основании выводов, сделанных о применении метода КББ, и исследований на их основе.

РАСЧЁТ МЕТРИК ПРОГРАММНЫХ СИСТЕМ НА ОСНОВЕ АНАЛИЗА КОДА И ИСТОРИИ ИЗМЕНЕНИЙ

Бурындин С.А. – студент, Тамплон А.В. – технический директор ООО «Энтерра-Софт»
Алтайский государственный технический университет (г. Барнаул)

Введение

В современном мире разработка ПО превратилась в одну из самых дорогостоящих индустрий и любые узкие места в технологическом процессе его создания могут привести к нежелательным результатам. Отсутствие адекватных инструментов, способных помочь менеджерам в ведении проектов, негативно сказывается на конкурентоспособности фирм. Для улучшения ситуации необходим инструмент, который будет отображать некоторые параметры проекта в процессе всей разработки. Инструмент, способный помочь менеджеру оценивать адекватность принимаемых решений и вклад отдельного программиста в разработку проекта.

Основной целью разрабатываемой системы является многомерный анализ результатов работы программиста. Результат работы программиста – это, разумеется, исходные тексты программ. Любая крупная компания, занимающаяся разработкой программного обеспечения, использует в своей практике различные инструменты, позволяющие программистам облегчить разработку и сопровождение проектов. Одним из таких инструментов являются системы контроля версий.

Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости, возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение и многое другое.

В подобных системах и хранится результат работы программиста. Рассмотрим основные проблемы, которые могут возникнуть при анализе такой информации.

1. Большой размер репозитария

Как уже было сказано, в системе контроля версий хранятся версий документов. Версии, или на языке SVN – ревизии, создаются регулярно в процессе разработки проекта. Поэтому репозитарий одного проекта получается очень большим и его полный анализ может занимать много времени. Следовательно, наша система должна также регулярно анализировать исходный ход, хранящийся в репозитарии, например, каждый день, а результаты анализа сохранять. Тогда за один раз необходимо будет анализировать гораздо меньший объем кода. Для этого система должна иметь возможность периодического запуска по расписанию. Кроме того, система должна сохранять промежуточные результаты.

Существуют различные методы хранения данных. Основные из них это файлы и базы данных. Первый метод сразу отвергаем, так как он имеет множество недостатков, о которых не раз упоминалось в различных источниках. Однако работа с объектно-ориентированным программным обеспечением и реляционными базами данных одновременно, доставляют программистам немало головной боли. И, кроме того, отнимают очень много времени на разработку приложений. Нам необходимо найти технологию сохранения данных, которая бы

удовлетворяла следующим требованиям:

- Легкость сохранения и последующей загрузки данных
- Возможность работы без предварительной настройки СУБД
- Возможность работы практически с произвольной СУБД

Всем вышеперечисленным требованиям удовлетворяет Hibernate. Данную технологию и было решено использовать при разработке системы.

2. Взаимодействие с SVN

Система должна уметь взаимодействовать с SVN. А именно: получать файлы произвольных ревизий и соответствующую дополнительную информацию. Это возможно либо посредством командной строки, либо с помощью специальных библиотек, которые предоставляют высокоуровневый интерфейс для взаимодействия с SVN. Первый способ не является попросту не удобным. В процессе поиска был обнаружен проект под названием «SVNKit» компании TMate Software. Компания разработала весьма удачную библиотеку с одноименным названием SVNKit. Её и было решено использовать.

Как мы уже решили, наша система будет запускаться регулярно, чтобы анализировать не все ревизии, а лишь непроанализированные. Алгоритм очевиден: необходимо сохранять последнюю проанализированную ревизию. Тогда при запуске система будет загружать информацию о последней проанализированной ревизии. Если такой информации нет, будет анализировать все ревизии с 0 до последней. Если такая информация есть, анализируем ревизии, начиная с последней сохраненной до ревизии HEAD – последней ревизии в репозитарии. Но и это ещё не все. Дело в том, что нам нет необходимости анализировать всю ревизию. Ревизии SVN создает всякий раз, когда программист выполняет команду commit. Весьма распространенный случай: программист подправил несколько файлов в проекте и сделал commit изменений. Тогда ревизия будет состоять из измененных файлов + файлы с предыдущей ревизии. Именно поэтому нам нет необходимости анализировать файлы, которые не изменялись. Сразу же возникает вопрос: а как узнать, какие файлы изменялись, а какие нет? Это очень легко. Каждая ревизия имеет некоторую служебную информацию, именуемую логом, которая содержит в себе:

- имя пользователя, который создал ревизию;
- дату создания ревизии;
- измененные файлы вместе с типом изменения: изменен, удален, добавлен и др.
- другую информацию.

Таким образом, реализуя данный алгоритм работы, наша система будет максимально использовать возможности SVN.

3. Расширяемость

Разрабатываемая система должна иметь возможность легко расширяться. То есть процесс добавления поддержки новых языков и метрик к существующим должен быть предельно простым.

Для построения легко расширяемого приложения необходимо найти способ, как просто и изящно добавлять к нашему проекту различные модули. Это необходимо по нескольким причинам.

Во-первых, это позволит разработать некое небольшое по объему ядро приложения, остальной функционал будет добавляться с помощью модулей. Это, например, дополнительные метрики, поддержка новых языков программирования и др.

Во-вторых, для небольших изменений достаточно будет изменить модули, не трогая ядро. И наоборот, можно будет целиком переписать ядро так, что остальные модули будут работать с новой версией ядра. Так называемая, обратная совместимость – новая версия ядра будет работать со старыми метриками.

В-третьих, модули могут писать независимые программисты, в то время как монолитное приложение разрозненно писать намного сложнее.

В-четвертых, предприятиям, которые будут использовать наш продукт, для добавления новой функциональности к существующей системе не придется целиком обновлять систему. Достаточно будет, добавить к существующей системе новый модуль.

В процессе поиска способов расширения был обнаружен проект под названием Java Plugin Framework (JPF).

Инфраструктура JPF основана на принципе плагинов. Плагин - структурный компонент, который соответствует коду и ресурсам системы и должен быть структурно описан. Эти плагины могут определять точки расширения, методы подключения, которые могут быть расширены другими плагинами.

Таким образом, используя Java Plugin Framework, мы можем сделать нашу систему модульной и легко расширяемой. При запуске приложения будут загружаться все необходимые модули, путь к которым будет указан в файле конфигурации.

Заключение

Таким образом, намечены способы решения основных проблем, которые могут возникнуть при разработке системы. Благодаря этому система будет иметь следующие возможности:

1. Система будет иметь возможность легко расширяться. То есть процесс добавления поддержки новых языков и метрик к существующим будет предельно простым.
2. Система будет взаимодействовать с системой контроля версий. А именно: получать файлы произвольных ревизий и соответствующую дополнительную информацию.
3. Система будет рассчитывать метрики и будет иметь возможность сохранять их.

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РАСЧЕТА ШТАТНОЙ ЧИСЛЕННОСТИ ПРОФЕССОРСКО-ПРЕПОДАВАТЕЛЬСКОГО СОСТАВА ВУЗА

Гвоздев А.С. – студент, Ананьев П.И. – ст. преподаватель
Алтайский государственный технический университет (г. Барнаул)

В рамках сложившейся кризисной ситуации в стране и в мире в целом можно наблюдать такое явление, как сокращение штатов сотрудников, вызванное неполной, либо неактуальной информацией о потребности в кадрах. Данный процесс неизбежен как в коммерческих предприятиях, так и государственных структурах, в том числе и ВУЗах, что доказывает важность контролирующей и прогнозирующей функций кадровой политики. Кроме того, в условиях проходящей реформы системы высшего образования, контроль и учет распределения нагрузки на кафедры ВУЗа является одной из приоритетных задач для обеспечения стабильного учебного процесса и повышения качества образования в целом.

Исходя из этих соображений, по заказу учебно-методического отдела Алтайского Государственного Технического университета им. И.И. Ползунова мной было разработано

программное обеспечение для расчета штатной численности профессорско-преподавательского состава вуза. Данный программный продукт разрабатывался как дополнительный интегрируемый модуль к уже существующей Единой Автоматизированной Информационной Системы университета, что позволило с одной стороны расширить функциональность уже существующей системы и предоставить пользователю требуемый продукт, при этом избежав затрат на покупку оборудования и ПО.

Было предложено сделать приложение на основе клиент-серверной архитектуры с одним сервером управления базами данных для обеспечения работы приложения одновременно у нескольких пользователей, чтобы вместе с тем данные сохраняли свою актуальность на всем протяжении использования программного продукта. В качестве сервера СУБД был выбран Oracle 10g. Для написания клиентской части приложения была выбрана среда Microsoft Visual FoxPro 9.0 SP 2. Данный набор ПО для разработки обусловлен хорошо известными механизмами взаимодействия этих продуктов, а также тем, что со времени их релиза прошло достаточное количество времени и основная часть ошибок уже была исправлена разработчиками, что позволит достичь высокого уровня надежности разрабатываемого приложения. В процессе разработки были использованы как средства встроенного языка программирования Microsoft Visual FoxPro, так и серверного PL/SQL, а также Java EE и SQLj для написания хранимых процедур архивирования данных.

В реализованном продукте пользователю будет доступна работа с необходимыми данными: списками кафедр, факультетов, специальностей, учебными планами по специальностям с учетом выбранной формы обучения. Также в продукте реализованы стандартные отчеты, форма которых была утверждена на этапе проектирования. Они позволяют пользователю выполнить следующие действия в процессе работы:

- Рассчитать штатную численность ППС ВУЗа на всех кафедрах
- Определить контингент учащихся к расчету штатной численности ППС ВУЗа
- Рассчитать распределение ставок по факультетам
- Рассчитать штатную численность ППС ВУЗа по кафедре
- Получить сводные данные по дисциплине
- Получить выписку из учебного плана по специальности
- Рассчитать распределение ставок на специальность
- Рассчитать дополнительный контингент на кафедрах ВУЗа

Кроме того, в системе предусмотрена возможность архивирования данных и работы с архивом, в том числе и генерация отчетов по архивным данным.

На данный момент разрабатываемый программный продукт находится на этапе внедрения и готовится к сдаче в эксплуатацию. В процессе внедрения была подтверждена полная совместимость программного продукта с уже существующей Единой Автоматизированной Информационной Системы университета, что говорит о качественной работе разработчика на этапе проектирования системы. На этапе тестирования продукт показал высокую устойчивость к сбоям в процессе работы.

Функциональная часть продукта позволит сотрудникам учебного отдела оперативно получать требуемые отчеты и при необходимости корректировать нагрузку на те, или иные кафедры. Экономия рабочего времени на этом этапе по моим расчетам составит около 20%, что позволит освободить время для выполнения других задач. Возможность архивирования данных позволит не только сэкономить время на заполнение данных на новый учебный период, но и проследить динамику использования штатов ВУЗа по кафедрам, факультетам, специальностям.

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА ДЛЯ РАСЧЕТА ДИНАМИКИ И КИНЕМАТИКИ КРИВОШИПНО-ШАТУННОГО МЕХАНИЗМА ДВИГАТЕЛЕЙ ВНУТРЕННЕГО СГОРАНИЯ

Говор А.В. – студентка, Кузьмин А.Г. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Роль динамических исследований в наши дни возрастает в связи с повышением уровня форсирования двигателей (при этом существенно возрастают нагрузки на звенья преобразующих механизмов) при одновременном снижении их металлоемкости (вследствие чего возникает проблема обеспечения прочности и жесткости этих звеньев).

Динамический расчет позволяет определить силы и моменты, воздействующие на детали двигателя, и является предшественником расчета их на прочность.

Цель данной работы – разработка программы для определения нагрузки на элементы кривошипно-шатунного механизма (КШМ) двигателей внутреннего сгорания (ДВС).

Для реализации данной работы была изучена предметная область и выполнены следующие этапы:

- ознакомление с принципом работы двигателя внутреннего сгорания (ДВС);
- изучение схем КШМ и вариантов компоновки двигателей;
- изучение уравнений, описывающих нагрузки на элементы КШМ;
- разработка алгоритма расчета.
- программная реализация алгоритма.

В качестве исходных данных задается геометрия двигателя, переменное во времени давление в цилиндре, массы подвижных деталей КШМ.

Разрабатываемая программа включает расчеты кинематики звеньев преобразующих механизмов и их отдельных точек; расчеты сил инерции, развиваемых этими звеньями, и разработку динамических моделей звеньев и механизмов в целом.

Производится расчет сил, действующих в кинематических парах, преобразование этих сил, определяется крутящий и набегающие моменты на шейки коленчатого вала, рассчитывается суммарная нагрузка на детали методом фазовых сдвигов с учетом порядка работы цилиндров и схемы КШМ.

Исходя из поставленной задачи, была спроектирована и реализована структура программного комплекса:

1. Слой данных, который содержит описание базовых классов данных, необходимых для работы всей системы;
2. Слой сеанса, который включает в себя функции по выполнению расчетов, используя базовые классы, а так же хранит информацию о состоянии и контексте работы;
3. Слой визуализации данных, который содержит методы для наглядного представления информации и текущего состоянии работы программы;
4. Слой пользовательского интерфейса, который предоставляет пользователю все необходимые средства для удобной работы.

Программный комплекс позволяет решать следующие задачи:

1. Ввод первоначальных данных, сохранение введенных данных, возможность выгрузки сохраненных ранее данных;
2. Сопровождение расчетов графической визуализацией;
3. Получение результатов расчетов в виде табличных данных с возможностью их сохранения на жестком диске;
4. Построение графических отчетов по результатам произведенных расчетов;
5. Просмотр графиков в любом масштабе;
6. Формирование различных отчетов в формате MS Excel и MS Word.

Таким образом, определяются условия, необходимые для последующих прочностных расчетов, что позволяет оптимизировать многие конструктивные особенности двигателя. Полученные результаты позволяют ускорить проведение динамических расчетов, а

графическое представление информации повышает наглядность и облегчает изучение предмета «Динамика ДВС».

ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ WEB-КЛИЕНТА ДЛЯ СИСТЕМЫ ENTERRA GIS

Гоменюк Р.В. – студент, Тамплон А.В. – технический директор ООО «Энтерра-Софт»
Алтайский государственный технический университет (г. Барнаул)

Уже ни для кого не секрет, что космические технологии стоят на службе простых людей, решая их повседневные задачи. Возможностями мобильной связи уже никого не удивишь, поэтому информационные и телекоммуникационные технологии решают уже новые задачи и потребности: мониторинг подвижных объектов с помощью системы спутниковой навигации.

Как работает спутниковая навигация? Global Positioning System (GPS) – это спутниковая навигационная система, состоящая из работающих в единой сети 24 спутников, которые постоянно движутся со скоростью около 3 км/сек, совершая два полных оборота вокруг планеты менее, чем за 24 часа. Система предназначена для мониторинга транспортных средств и позволяет определять местоположение подвижных объектов с точностью до 1 метра в режиме реального времени. Принимая сигналы от спутников, навигационный прибор может рассчитать скорость и направление Вашего движения. Но для обработки поступающих со спутников данных и представления их пользователю в удобном формате необходимо специальное программное обеспечение – так называемый «посредник» между Спутником GPS и пользовательским ПК. Такую систему-посредника в Сибири предлагает компания Энтерра. Система Enterra GIS сделала возможным управление целым автопарком с одного ПК (быстро и легко). Схема работы системы EnterraGIS представлена на рисунке 1.

Рисунок 1 – схема работы системы EnterraGIS

Пользователи взаимодействуют с системой EnterraGIS через специализированное программное обеспечение, которое устанавливается на их компьютеры. Программное обеспечение выполняет следующие функции:

- объективный и оперативный контроль движения средств транспорта с помощью

взаимодействия с системой EnterraGIS;
- отображение текущих данных о состоянии средств транспорта;
- отображение истории передвижения транспортного средства за определенный промежуток времени;
- добавление в систему EnterraGIS новых маршрутов, контрольных точек и зон, для контроля движения транспортных средств.

В процессе использования системы были выявлены несколько недостатков такого подхода, а именно:

- Необходимость в частом обновлении программного обеспечения. Системы развивается, добавляются новые возможности, выходят новые версии программного обеспечения, следовательно пользователям необходимо постоянно обновлять свои программы.
- Т.к. взаимодействие пользователей с системой идет через ПО, установленное на компьютеры пользователей, то воспользоваться возможностями системы на других компьютерах нельзя. Т.е. пользователи «привязаны» к своим компьтерам.
- Зависимость программного обеспечения от операционной системы. Т.к. программное обеспечения разработано для работы в Microsoft Windows, то использование его в других ОС практически невозможно.

Цель данной работы – устранение основных недостатков клиентской части системы EnterraGIS. Решением, устраняющим недостатки текущей клиентской части, было выбрано проектирование и реализация web-клиента для системы EnterraGIS. Web-клиент представляет собой web-сайт, зайдя на который пользователи получают доступ ко всем основным функциям системы EnterraGIS. Очевидно, что данное решение устраняет основные недостатки текущего программного обеспечения.

Т.к. многие пользователи работают на текущем ПО и их все устраивает в его работе, то необходимо оставить возможность использования этого ПО. Т.о. новая архитектура системы EnterraGIS имеет следующий вид:

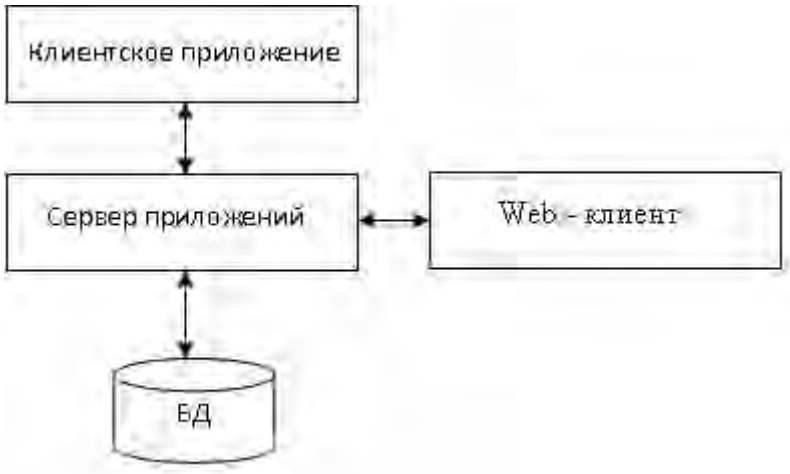


Рисунок 2 – архитектура системы EnterraGIS

Новая архитектура позволяет пользоваться возможностями системы EnterraGis как через клиентское ПО, так и через web-клиент с любого компьютера, подключенного к глобальной сети Интеренет. Взаимодействие клиентских приложений с сервером приложений осуществляется на основе протокола TCP/IP. Сервер приложений хранит очередь запросов. При поступлении нового запроса от клиентского приложения, этот запрос заносится в очередь запросов. Сервер обслуживает очередь запросов в порядке их поступления.

Реализация web-клиента ведется с использованием современных технологий:

1. Microsoft .Net Framework 3.5 – реализация модуля взаимодействия с сервером приложений;
2. Microsoft Asp.Net 3.5 – реализация интерфейса web-клиента;
3. Microsoft Asp.Net AJAX – реализация интерфейса web-клиента.

РАЗРАБОТКА ПОИСКОВОЙ СИСТЕМЫ ДЛЯ WEB-РЕСУРСА АКУНБ ИМ. В.Я. ШИШКОВА

Горбунов А.А. – студент, Лукоянычев В.Г. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

На сегодняшний день трудно себе представить работу в офисе без использования оргтехники. В каждой организации сотрудники ежедневно сталкиваются с задачами связанными с копирование либо распечатыванием документов. Но ресурса картриджей в оргтехнике, при интенсивной работе, хватает на весьма ограниченное количество времени. В связи с чем возникает вопрос: покупать новый картридж, либо заправлять пустой. Как правило, в организациях принимается решение заправлять пустой картридж, так как этот вариант заметно выгоднее в экономическом плане.

На данный момент в Алтайском государственном техническом университете им. И.И. Ползунова принято решение все действия по заправке картриджей для принтеров и другой оргтехнике производить централизовано. И для того, чтобы помочь работникам пункта приема заявок на заправку картриджей, мною было написано web-приложение для обработки заявок.

Клиент-серверная архитектура системы основана на использовании web-технологий организации рабочего места, что позволяет объединить в единую модель не только отдельно взятые структурные подразделения АлтГТУ, но и в дальнейшем поможет при организации доступа из глобальной сети internet.

Функционал системы позволяет решить все типовые задачи пункта приема заявок:

- автоматизировать регистрацию документов и заданий;
- обеспечить эффективное взаимодействие сотрудников в рамках работ по документам;
- осуществлять мгновенный поиск информации;
- контролировать выполнение работ, инициируемых документами и заданиями;
- проводить мониторинг состояния выполняемых процессов за счет формирования различных журналов и отчетов;
- организовать долговременное хранение документов организации;
- обеспечить разграничение прав доступа сотрудников к информации.

Основные преимущества системы:

- Простое решение. Интуитивно понятный интерфейс не требует обучения сотрудников.
- Web решение. Система доступна круглосуточно с любого компьютера, имеющего установленный браузер и доступ к сети.
- Безопасность, обеспеченная защищенным каналом передачи данных, отсутствием физического доступа к серверу, резервным копированием базы данных, разделением прав доступа сотрудников.
- Не нужно устанавливать программу на компьютеры пользователей.

Благодаря модульной структуре приложения, в дальнейшем будет просто внести дополнения к функциональности. Ближайшим действием по доработке системы предусматривается организация доступа из глобальной сети internet, что позволит следить за системой из любой точки планеты при условии наличия компьютера с доступом в интернет.

РАЗРАБОТКА ВЕКТОРИЗАТОРА ДЛЯ ОБРАБОТКИ ПОЛНОЦВЕТНЫХ КАРТОГРАФИЧЕСКИХ ИЗОБРАЖЕНИЙ

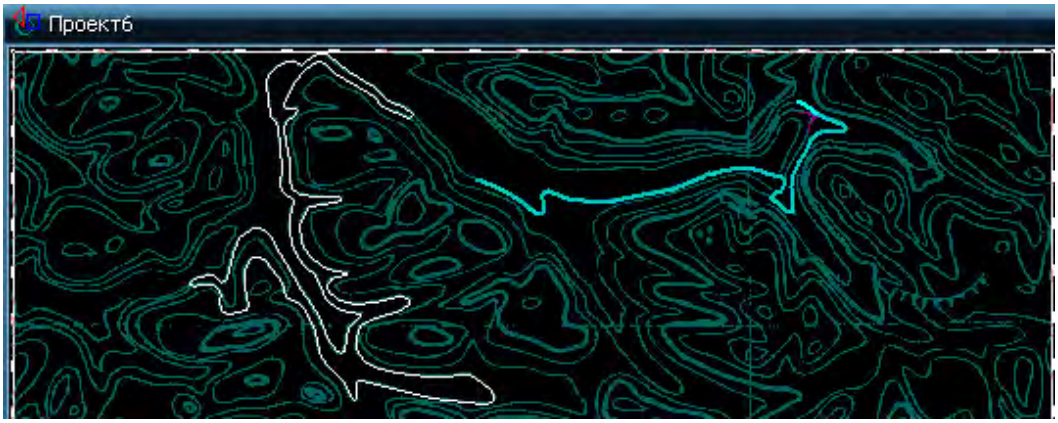
Горлач О.В. – студент, Андреева А.Ю. . – к.ф.м.н., доцент
Алтайский государственный технический университет (г.Барнаул)

Векторизация — перевод из растровой графики в векторную, в отличие от растеризации (перевод в растровый формат), этот процесс очень сложный и не может быть выполнен полностью в автоматическом режиме. Соответственно векторизаторы это программы выполняющие векторизацию растровых изображений таких как карт, чертежей, и другой полезной графической информации находящейся в растровом виде.

До недавнего времени средства обработки человеком картографической информации оставались практически неизменными. Для понимания и анализа географической информации использовались бумажные карты, которые первоначально рисовались от руки, затем стали изготавливаться типографскими способами. Тем не менее, данный способ далек от идеала. Будучи однажды нарисованной или напечатанной, карта со временем теряет свою актуальность, и для того, чтобы карта была действительно полезной для человека, его использующего, требуется заменять ее время от времени на более «свежую», содержащую новые, появившиеся на изображаемой области, объекты, а также отредактированные изображения объектов, изменивших свою форму и положение. Картографическая информация в настоящее время широко используется в геоинформационных системах, поэтому перспективно иметь возможность переводить существующие карты в векторный формат и в случае необходимости дополнять их, так как составление новых карт является очень дорогостоящим и длительным процессом. В настоящее время существует множество программ выполняющих векторизацию изображений в том числе и картографических, ниже приведён краткий обзор двух программ EasyTrace и AcmeTraceArt.

Обзор программы EasyTrace

Программа EasyTrace предназначена для переноса графической информации с бумажных носителей в компьютер и ориентирована, прежде всего, на обработку картографических материалов. Векторизатор Easy Trace работает в двух режимах трассировки – ручном и автоматическом. В автоматическом режиме чтобы начать трассировку сплошной или пунктирной линии, нужно указать на изображение точку затравки на «хорошем» участке, где для трассировщика не предвидится осложнений. Для начала трассировки точечной линии нужно последовательно указать две соседние точки, задав, таким образом, примерный шаг и направление. После ввода информации трассировщик прослеживает дальнейшее направление линий и в случае нескольких вариантов предлагает выбрать подходящее направление. В ручном режиме процесс векторизации необходимо выполнять пользователю проходя участки которые программа не может распознать. На рисунке ниже показан результат векторизации, на котором линиями белого цвета показаны ломаные линии выявленные векторизатором. Линиями голубого цвета показана распознаваемая линия, а линиями фиолетового цвета показаны варианты дальнейшего направления линии так как на данном участке программа не может точно определить направление кривой.



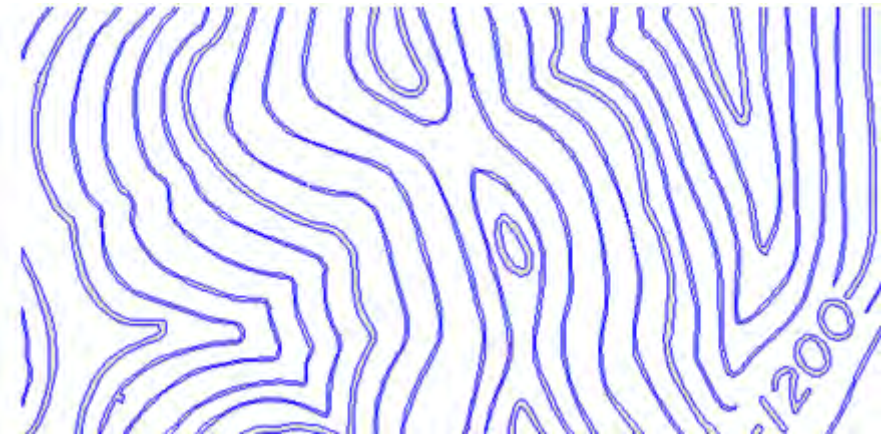
Обзор программы Acme TraceArt

Trace Art многофункциональная и сложная программа, предназначенная в основном для векторизации чертежей, схем и разного рода карт. Поддерживается большое количество растровых и векторных форматов. TraceArt так же имеет возможности настройки параметров векторизации как:

- очерчивание линий по контуру;
- очерчивание линий по центру каркаса;
- распознавание линий по шаблону (т.е. векторизация вертикальных или горизонтальных линий).

Программа работает в автоматическом режиме используя алгоритмы векторизации бинарных изображений и не может работать с многоцветными растрами.

На рисунке ниже представлен результат векторизации картографического изображение с выделением линий по контуру.



Рассмотренные программы для векторизации растровых изображений обладаю довольно различной функциональностью, способами векторизации (ручной, полуавтоматической и автоматической), имеют различные настройки процесса векторизации так как при базовой конфигурации практически невозможно получить необходимые результаты трассировки. Но главным недостатком рассмотренных решений задачи векторизации является отсутствие возможности производить векторизацию многоцветных растров. Хотя в программе EasyTrace и имеется возможность полуавтоматической векторизации, но при растровом изображении с большим числом цветов автоматизм программы теряется, и она не может корректно выполнить дальнейшую доводку объектов, в результате вся работа по векторизации ложится на пользователя в ручном режиме.

Так как в настоящее время широкое распространение получили многоцветные карты поэтому имеется потребность в возможности перевода данных растровых карт в векторный формат.

В рамках решения поставленной задачи реализованы следующие возможности:

- реализована сегментация многоцветных изображений с помощью алгоритмов «Волшебная палочка», «Умные ножницы», «GraphCut», для выделения связанных областей на карте и возможностью настройки пользователями параметров методов;
- отслеживание граничных линий на растре, разделяющие пиксели растра на области, окрашенные в одинаковые цвета;
- аппроксимация выделенных граничных линий кривыми Безье, с помощью B-сплайнов и бета- сплайнов, с возможностью настройки пользователями параметров аппроксимации;
- послойное хранение полученных кривых и линий контура, с возможностью переноса объектов из одного слоя в другой;
- функции экспорта полученного векторного изображения в наиболее распространенные форматы DFX, SVG, формат обмена данными MapInfo(MIF и

MID);

– редактор векторной графики позволяющий пользователю редактировать полученное в результате векторизации векторное изображение и создавать собственные объекты такие как линии, кривые, многоугольники, эллипсы, площадные объекты.

О НЕКОТОРЫХ ВОПРОСАХ РАЗРАБОТКИ ПЛАГИНА КОНВЕРТАЦИИ ВХОДНОГО АУДИОПОТОКА ПРИ ЗАПИСИ В РЕЖИМЕ РЕАЛЬНОГО ВРЕМЕНИ ДЛЯ ВИРТУАЛЬНОЙ ЗВУКОЗАПИСЫВАЮЩЕЙ СТУДИИ

Гринштейн С.С. – студент, Сорокин А.В. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

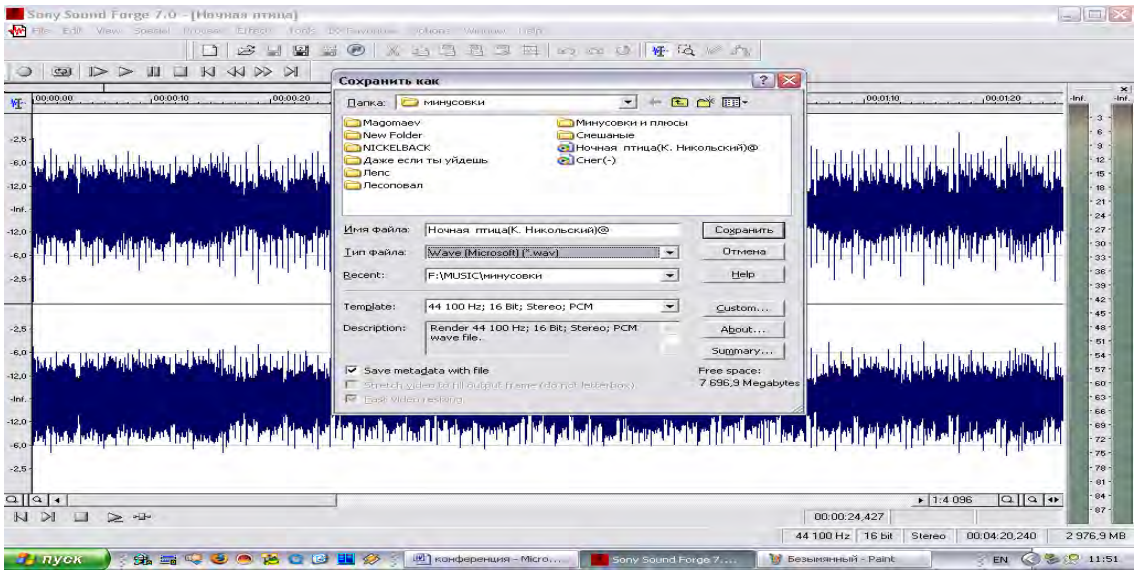
Введение

В настоящее время существует большой список программ, позволяющих конвертировать музыкальные файлы из одного формата в другой (Arial AudioConverter v2.3.58, SuperAVConverter v9.1, Super v2008 и т. д.). Возможность аудио конвертации поддерживают многие программные аудиоплейеры (FreeRip MP3 v3.05, Music Man v 5.12.110 и т.д.) и специализированные программные продукты для работы с музыкальными файлами, такие как виртуальные студии звукозаписи и аудиоредакторы (Cakewalk Sonar, Steinberg Cubase SX, Sound Forge, Adobe Audition и т.д). Таким образом, конвертеры могут быть как отдельными программами, так и встроенными в другие программы или на этапе их разработки, или на этапе их обновления посредством «плагинов»(Plug-In).

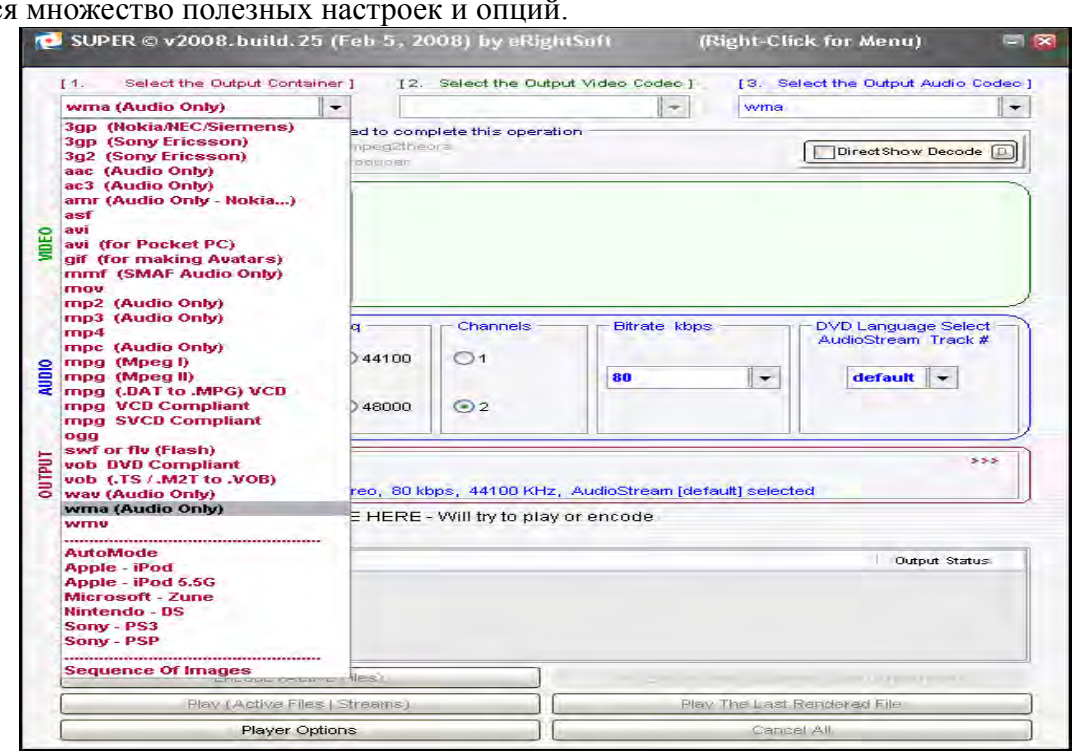
Конвертация (кодирование) происходит с помощью специальных программ – кодеков. Эти программы способны выполнять преобразование музыкальных файлов из одного формата в другой. При этом можно указать различные параметры, которые влияют на качество кодирования, а также на объем выходного файла.

Рассмотрим три существенно отличающихся программных продукта: популярный аудиоредактор Sound Forge 7.0, конвертер Super v2008, программа для конвертации в реальном времени Total Recorder 7.0.

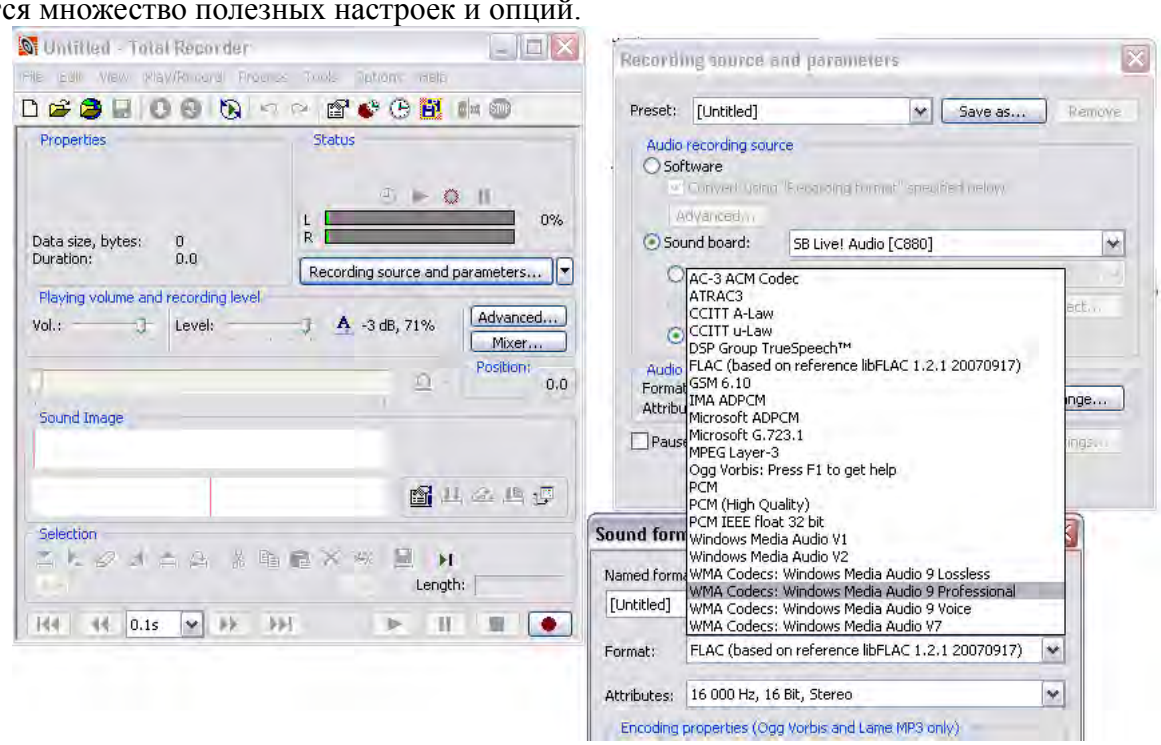
Sound Forge 7.0 позволяет сохранять любой открытый музыкальный файл, либо записанный в этой же программе свой музыкальный файл в формат, выбираемый из специального списка, который появляется при выборе пункта меню Save As (Сохранить как). Этот список строится на основе доступных программе кодеков на данный момент. Список можно расширить, используя дополнительные кодеки в виде плагинов.



Super v2008 позволяет конвертировать видео- ,аудиоданные в огромное количество различных форматов. Программа имеет систему всплывающих подсказок, которые помогают быстрее разобраться в том, как она функционирует. Имеется встроенный проигрыватель. Имеется множество полезных настроек и опций.



Total Recorder 7.0 позволяет сохранять любой открытый музыкальный файл, либо записанные в этой же программе свои музыкальные файлы в формат, выбираемый из специального списка. Также есть возможность конвертации во время записи или воспроизведения в режиме реального времени. Причем программа может перехватывать и конвертировать аудиопоток, который воспроизводится какой-либо другой программой. Имеется множество полезных настроек и опций.



Все рассмотренные выше программные продукты не обладает требованиями, описанными ниже:

- Программы не могут быть представлены в форме плагина;
- Программы не могут перехватывать и конвертировать входной аудиопоток, который записывается какой-либо другой программой;
- Программы не принадлежат к классу бесплатного распространения ПО.

В связи с вышесказанным возникает задача создания программного продукта, отвечающего всем указанным выше требованием, свойства которого приведены в следующем разделе.

Постановка задачи

Предметом исследования данной работы являются стандартные подсистемы операционной системы типа Windows NT для работы со звуком такие как:

- подсистема сжатия звука (Audio Compression Manager - ACM, диспетчер сжатия звука).
- подсистема мультимедийного (звукового, видео и им подобного) файлового ввода/вывода **MMIO** (Multimedia Input/Output).
- микшерная подсистема

В работе рассматривается задача разработки программы, осуществляющей следующие действия:

- захват записываемого аудиопотока с устройства записи
- преобразование аудиопотока в определенный заранее формат
- сохранение преобразованного аудиопотока в постоянную память по окончании записи.
- считывание сохраненного файла в определенном ранее формате и выполнение обратного преобразования
- загрузка файла в студию и его воспроизведение

Входными данными программы являются:

- захватываемый аудиопоток
- формат, в который нужно преобразовать захватываемый аудиопоток
- файл, закодированный в определенном заранее формате.

Выходными данными являются:

- файл, закодированный в определенном заранее формате.
- файл, в формате WAVE PCM 44.1 kHz

Требования предъявляемые к проектируемому программному продукту:

- правильность алгоритма
- бесплатное распространение
- полезность
- простота эксплуатации
- расширяемость
- поддержка наиболее распространенного стандарта для плагинов (например, VST, RTAS, VSTi), то есть совместимость.

Необходимостью разработки проектируемого программного продукта является следующее:

- Данная программа позволит значительно сэкономить затраты памяти на хранение записанных на студии звуковых файлов.
- Выбор произвольного формата конвертирования дает возможность выбора между качеством звука и объемом занимаемой памяти.

- Бесплатное распространение и расширяемость делают программу более доступной и полезной, чем подобные платные программы.

Создание программы будет производиться в форме плагина, что позволит встроить ее во множество различных студий, поддерживающих соответствующие стандарты, соглашения и платформы.

Методы решения задачи

Первый метод. В нем написание приложения производится с использованием интерфейсов таких подсистем Windows, как подсистемы сжатия звука (Audio Compression Manager - ACM, диспетчер сжатия звука) и подсистемы мультимедийного (звукового, видео и им подобного) файлового ввода/вывода **MMIO** (Multimedia Input/Output). Такое приложение потребует установленной операционной системы семейства Windows NT, либо ее эмуляции.

Второй метод. В данный момент находится в процессе исследования.

В нем написание приложения производится для аудиодрайвера ASIO с использованием библиотеки ASIO C++ Library. У данного драйвера выше производительность, чем у драйверов подсистем Windows, но тогда для работы приложения будет обязательно наличие данного драйвера.

Плагин будет разрабатываться по стандарту VST. Существует VST SDK, с помощью которого это можно выполнить. Также существуют программы, позволяющие разработать VST-плагин визуальными средствами, например SynthEdit, SynthMaker.

Список литературы

1. Музыченко Е. Подсистема сжатия звука в Windows. // Компьютер Пресс, №7, 2000. (http//www.compress.ru/Article.asp?id=658)
2. Музыченко Е. Низкоуровневое программирование звука в Windows. // Компьютер Пресс, №6, 2000. (http//www.compress.ru/Article.asp?id=249)
3. Музыченко Е. Обработка звуковых файлов в Windows. //Компьютер Пресс, №8,2000. (http//www.compress.ru/Article.asp?id=529)

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ОБРАБОТКИ ИНФОРМАЦИИ ПРИ ОРГАНИЗАЦИИ НАУЧНО-ТЕХНИЧЕСКОЙ КОНФЕРЕНЦИИ

Джабраилова Ф.С. – студентка, Сучкова Л.И. – к.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В настоящее время при организации проведения научно-технических конференций должна быть предусмотрена автоматизация сбора и обработки заявок на участие в конференции, а также формирование различной документации оргкомитета.

Разработан программный продукт, позволяющий с помощью Интернет-технологий упростить и ускорить взаимодействие участников и организационного комитета конференции, обеспечить хранение, а также обработку информации, связанной с проведением научно-технической конференции. Для хранения и обработки информации используется клиент-серверная СУБД MySQL, которая физически располагается на том же сервере, где и WEB-сайт. С использованием разработанного программного продукта участник может самостоятельно указывать всю необходимую оргкомитету информацию о своей заявке на участие, посланных им докладах, заказах сборника материалов конференции и т.п., а оргкомитет на правах администратора системы может информировать участников о получении и результатах рассмотрения докладов, о получении оргвзносов, о рассылке

сборников.

В рамках решения поставленной задачи реализованы следующие возможности:

1. Спроектирована структура базы данных для хранения всей необходимой в работе оргкомитета информации;

2. Разработан программный комплекс, состоящий из интерактивного WEB-сайта конференции и программного модуля для администрирования.

WEB-сайт научно-технической конференции позволяет участнику конференции осуществлять следующие возможности:

- регистрация участника на сайте при внесении интересующей оргкомитет информации об участнике и организации, которую он представляет;
- идентификация участника по регистрационному имени и паролю;
- отправка доклада с указанием информации о докладе, а именно, его названия, авторов доклада, раздела конференции, к которому относится доклад;
- заказ сборников конференции, в котором необходимо указать информацию о количестве сборников, количестве страниц, посланных участником конференции, адрес, по которому отправлять сборник;
- расчет стоимости сборника после заказа его участником;
- возможность загрузки ранее введенных данных для просмотра или редактирования.

Результаты работы сайта сохраняются на WEB-сервере в базе данных конференции.

Модуль администрирования осуществляет ведение разработанной БД оргкомитета конференции. В модуле реализованы следующие функции:

- добавление, редактирование и удаление всей имеющейся в базе данных информации;
- формирование ведомости для бухгалтерии, которая должна содержать название организаций, фамилии возможных плательщиков и сумму ожидаемого платежа;
- расчет оргвзносов;
- формирование списка авторов конференции;

Разработанный комплекс будет использован при проведении 10 Международной конференции «Измерение, контроль, информатизация».

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ПРОВЕРКИ ПРОЕКТНЫХ РЕШЕНИЙ НА СООТВЕТСТВИЕ НОРМАТИВАМ ПО ТЕПЛОЗАЩИТЕ ЗДАНИЙ

Камышев А.А. – студент
Шелудько Н.Н. – гл. инженер проекта ЗАО «ЗАПСИБНИИПРЕКТ»
Алтайский государственный технический университет (г. Барнаул)

В данной работе рассматривается описание возможностей программы, имеющей название: «Система проверки проектных решений на соответствие нормативам по теплозащите зданий», отличной от известных аналогов в данной области.

Программа предназначена для расчёта энергоэффективности здания по его проекту и проверки проекта на соответствие требованиям по теплозащите. Программа должна автоматизировать выполнение ввода данных, расчётов и составление энергетического паспорта здания.

Область применения: проектирование новых и реконструкция существующих зданий с нормируемой температурой и относительной влажностью внутреннего воздуха. Программа будет применима только для зданий, расположенных на территории Алтайского края.

Вследствие особенностей климата на большей части территории страны человек проводит в закрытых помещениях до 80% времени. Для создания нормальных условий его

жизнедеятельности необходимо поддерживать в этих помещениях строго определённый тепловой режим. Кроме того, некоторые технологические процессы требуют поддержания определённых климатических параметров, без которых невозможно их осуществление и которые способствуют получению продукции высокого качества.

Исходя из технико-экономической целесообразности комфортные условия должны поддерживаться не во всём объёме помещения, а лишь в местах, где человек живёт, трудится или отдыхает. Место преимущественной деятельности человека в условиях помещения называется обслуживаемой или рабочей зоной. Системы отопления, вентиляции и кондиционирования воздуха совместно с теплозащитой ограждений должны обеспечить расчётные тепловые условия в обслуживаемой зоне помещения [1].

Тепловой режим помещения характеризуется температурой внутреннего воздуха t_{int} ºС и температурой внутренних поверхностей τ_{int} ºС и считается комфортным (то есть расчётные тепловые условия обеспечиваются), если соблюдаются первое и второе условия комфортности. [2]

По первому условию комфортности поддерживается такой температурный режим в помещении, при котором человек, находясь в середине помещения, не испытывает перегрева или переохлаждения.

Второе условие комфортности определяет температурный режим для человека, находящегося около нагретых или охлаждённых поверхностей в рабочей зоне (главным образом в условиях производственных цехов).

Кроме задачи обеспечения комфортности в настоящее время особенно важной представляется задача оптимизации расходов предприятия и, в этом контексте, экономии расходов на обогрев помещения.

Наружные ограждения любых зданий должны предохранять помещения от непосредственных атмосферных воздействий и, в независимости, от свойств материала из которого они выполнены, от потери тепла через них и проникания холодного наружного воздуха в зимний период . В общей задаче создания комфортных для человека или оптимальных для технологического процесса условий эти функции ограждений оказываются очень важными.

Для определения теплового режима зданий, теплопотерь зданий была разработана соответствующая методика, представленная в ТСН Алтайского края [3]. С её помощью определяют соответствует ли конструкция здания и, в первую очередь, его ограждающие конструкции нормативным требованиям.

Наличие методики расчёта, каталогов строительных материалов, в которых указаны значения необходимых для выполнения расчётов параметров материалов, а также необходимость выполнения этих расчётов для значительного процента разрабатываемых и реконструируемых зданий в сочетании с их сложностью для выполнения человеком подводят к решению данной проблемы путём автоматизации расчётов с использованием ЭВМ.

В ходе проведённого исследования рынка ПО, были обнаружены следующие программные продукты, предназначенные для использования в указанной предметной области:

- СИТИС: Трак 2.0
- Рок'08
- Энергопаспорт'2007

Кратко опишем декларируемые разработчиками характеристики этих программ и выявленные в ходе использования демонстрационных версий их преимущества и недостатки.

СИТИС: Трак 2.0

Программа «**СИТИС: Трак**» предназначена для расчета теплотехнических параметров стен и перекрытий в соответствии с методикой расчета, приведенной в СП 23-101-2004, СНиП 23-02-2003, СНиП 23-01-99*, СТО 00044807-001-2006.

Возможности, преимущества и недостатки:

- вычисление всех расчётных параметров однородных перекрытий и стен; программа проста в использовании, однако выполняет лишь часть расчётов указанных в ТСН, не упрощает получение исходных данных по чертежам, не составляет итоговый отчёт — энергетический паспорт здания.
- Демонстрационная версия не стабильна, вычисления не производятся из-за возникающих ошибок, работоспособность полнофункциональной программы таким образом находится под вопросом.

Рок'2008

РОК ‘2008. Программа расчета теплотехнических характеристик ограждающих конструкций зданий и сооружений.

Программа позволяет рассчитать полный комплекс параметров здания необходимых для выполнения проверки соответствия здания нормативам по теплозащите. Тем не менее, данный программный продукт как и предыдущий предназначен для выполнения расчётов отдельных ограждающих конструкций и отдельных параметров проекта, и не рассматривает их как единое целое, и не составляет итогового отчёта, таким образом выполняя лишь часть работ.

Энергопаспорт'2007

Программа **Энергопаспорт '2007** предназначена для выполнения расчета параметров энергопаспорта жилых и общественных зданий и сооружений, формирования шаблонов раздела энергоэффективность.

Дополнительно реализован расчет параметров энергопаспорта для жилых зданий со встроенными помещениями общественного назначения.

Программа формирует отчет по энергопаспорту согласно приложению Д и протокольный отчет согласно приложению Г СНиП 23-02-2003. Также программа позволяет сформировать шаблоны пояснительных записок в формате MS Word по разделу Энергоэффективность (для общественных и жилых типов зданий).

Преимущества:

- программа обладает более широкими возможностями по выполнению расчётов: выполняет полный комплекс расчётов, включая расчёт смешанных зданий с жилыми и общественными помещениями. Составляет энергетический паспорт и пояснительную записку, экспортирует результаты в MS Word.

Недостатки:

- программа требует ввода входных данных пользователем и не поддерживает получение данных по чертежам;
- оформление пояснительной записки отличается от принятого в организации Заказчика.

Таким образом, ни одна из представленных программ не соответствует полностью требованиям заказчика. Также отсутствуют программы, которые бы обладали возможностью работы с проектной документацией, представленной в виде чертежей, то есть все программы требуют проводить анализ чертежей проектировщиком вручную. Учитывая, что именно анализ чертежей занимает наиболее значительное время, иногда 15 часов и более, то использование данных программных продуктов не приносит значительной выгоды, и их приобретение нецелесообразно.

Разработанный программный продукт позволяет выполнять расчёт по методике, которая описана в ТСН 23-325-2001 Алтайского края, с учётом изменений внесённых в неё заказчиком. При этом данные могут вводиться одним из двух способов:

- вручную, то есть проектировщик вводит все объёмно-планировочные параметры, климатические параметры и прочие при этом либо используются стандартные,

предложенные программой, либо вводятся также проектировщиком;

- автоматически — проектировщик предоставляет программе чертёж, далее программа анализирует его и получает все данные автоматически. Проектировщик во время выполнения анализа чертежа и по его завершению может корректировать предложенные программой значения параметров.

По результатам выполненного расчёта и проверки на соответствие нормативным параметрам, если проверка прошла успешно, то программа сообщает проектировщику класс энергетической эффективности здания и составляется отчёт, который содержит:

- энергетический паспорт здания,
- пояснительную записку.

Если результаты проверки не были положительными, то программа сообщает проектировщику, какие параметры здания не соответствуют требованиям.

Входные, расчётные данные и результаты и отчёты по всем проектам хранятся в базе данных программы. Там же хранятся таблицы нормативных и справочных данных, используемых при расчётах: проектировщик может в любой момент изменить такие справочные данные, как, например, варианты функциональных назначений зданий; нормативные параметры также можно изменить войдя в программу как администратор (на деле это может выполнить не только системный администратор, но и проектировщик, владеющий основами работы с ПК).

Программный продукт позволяет сократить время, необходимое на проведение теплотехнических расчётов, в несколько раз, повысив при этом качество работы, за счёт автоматизации вычислений объёмно-планировочных параметров, и освобождает проектировщика от рутинных работ в пользу принятия стратегических решений.

РАЗРАБОТКА ТЕХНОЛОГИИ И ИНСТРУМЕНТАРИЯ ДЛЯ СОЗДАНИЯ РАСПРЕДЕЛЁННЫХ СИСТЕМ НА ОСНОВЕ УНИФИЦИРОВАННОЙ СТРУКТУРЫ КАРКАСОВ СИСТЕМ И ПРИЛОЖЕНИЙ

Князьков К.В., Карымов И.Л. – студенты, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Распределённые системы являются принципиально отличным классом программных комплексов, способных решать задачи, с которыми системам других архитектур справиться не под силу.

Построение распределенных систем нельзя назвать новой задачей. В зависимости от круга решаемых задач и расстановки акцентов на различные особенности системы, можно выделить целый ряд существующих подходов, каждый из которых обладает своими преимуществами и недостатками:

1. Использование низкоуровневых языков программирования (C, Fortran) и библиотек передачи сообщений (PVM, MPI). Обеспечивает высокую производительность, поэтому широко применяется для сложных вычислительных расчетов.
2. Распределенные системы объектов (CORBA, DCOM, Globe), объединяют в себе преимущества распределённой архитектуры с гибкостью и формализмом объектно-ориентированного подхода, позволяют программам на различных языках работать с единым пространством объектов. [1]
3. J2EE — технология, основанная на компонентном подходе к сервисно-ориентированной архитектуре.
4. Специализированные языки программирования: Occam, Erlang. Дизайн таких языков изначально исключает многие проблемы параллельного и распределенного

программирования. [2]

5. GRID - технология-тяжеловес, которая по оценкам специалистов, в перспективе позволит обеспечить возможность создания распределенных вычислительных систем сверхвысокой пропускной способности, позволяющих решать глобальные стратегические проблемы человечества.
6. ZeroC ICE - объектная система, обеспечивающая реализацию промежуточного слоя в распределенных приложениях. [3,4]
7. Решения, реализующие концепцию облачных вычислений: Amazon EC2 (Elastic Cloud), GAE (Google Application Engine), Microsoft Azure Services. [5]

Отметим, что, несмотря на большое количество исследований по данной тематике остается ряд актуальных проблем, связанных как с самими технологиями, так и с недостатками конкретных их реализаций. Большинство подходов представляют собой крайности в выборе между решением узкого круга специфических задач и попыткой охватить все возможные сферы применения и, как следствие этого, либо чрезвычайно сложно адаптируются под особенности конкретного проекта, либо громоздки и сложны в освоении и использовании.

В последнее время рост производительности, снижение стоимости сетевого оборудования и аппаратной части обычных компьютеров привели к тому, что всё более широкий круг программистов (небольшие компании) получает возможность создавать распределённые системы для эффективного решения повседневных задач. Зачастую, существующая инфраструктура организации среднего размера (например, вуза), без каких-либо модификаций, уже может стать базой для создания мультикомпьютерной гетерогенной среды. Однако, для того, чтобы распределенные системы смогли окончательно выйти за стены лабораторий и больших корпораций, необходимо появление нового поколения инструментов и технологий, которые изменили бы облик процесса их разработки.

Такая технология должна удовлетворять определенным требованиям: позволить программисту сосредоточится на написании функциональной части приложения, не отвлекаясь при этом на решение таких сложных моментов реализации распределенных систем, как масштабируемость, репликация, параллельная обработка, балансировка нагрузки, учет реальных физических условий (отказ оборудования и программного окружения, адресации узлов, топология сети).

Авторами предлагается подход и реализация соответствующего инструментария, которые позволяют обеспечить перечисленные требования. Подход заключается в предоставлении разработчику каркаса программной системы, реализующего универсальную архитектуру распределенного приложения. Предлагаемая архитектура основана на модели распределенной программной системы, описанной ниже.

Распределенная программа представляет собой структуру взаимодействующих модулей. Под модулем далее будем понимать некую абстракцию (см. рисунок 1), характеризуемую:

1. Описанием своих интерфейсов и внешних вызовов.
2. Шаблоном поведения.
3. Функциональным кодом, состоящим из активной части и реализации его интерфейсов (пассивной части).

В общем случае, модуль объединяет в себе активную часть, инициирующую некоторые действия, и пассивную, отвечающую за интерфейсные методы. В случае отсутствия одной из частей, он вырождается в клиент либо в сервер соответственно (в терминах клиент-серверной архитектуры).

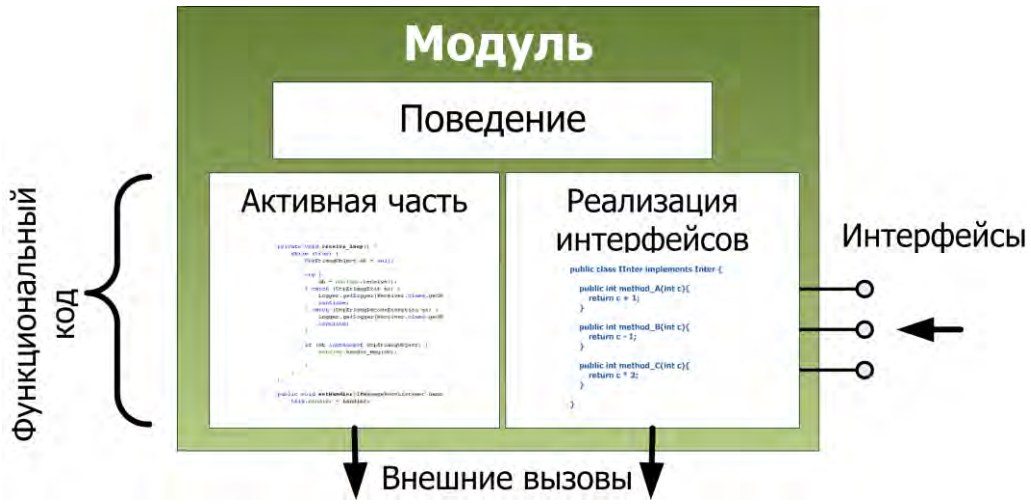


Рисунок 1. Модуль

Поведение - это набор правил, по которым система исполнения работает с модулем. Выбор поведения возможен из определенного числа шаблонов:

1. Масштабируемость — возможность запуска дополнительных экземпляров модуля.
2. Реплицирование — необходимость создания пассивных копий модуля (с поддержкой непротиворечивости их состояний) для обеспечения замены экземпляра модуля в случае его отказа.
3. Сериализуемость — возможность сохранения текущего состояния модуля.
4. Переносимость — возможность переноса модуля в распределенной среде с сохранением его внутреннего состояния.
5. Дублируемость — возможность тиражирования экземпляра модуля с сохранением его внутреннего состояния.

Каждый модуль является классом для своих экземпляров (в терминах ООП - отношение класс-объект). В зависимости от определенного для него поведения, в процессе выполнения программы, он может иметь различное количество экземпляров. Взаимодействие модулей возможно только на уровне классов, т.е. модуль может вызвать метод или послать сообщение только классу, а не конкретным его экземплярам. Если у модулей определено поведение, то есть возможность воспользоваться специфическими для него типами вызовов. Пример

Предположим, что в модуле B реализован интерфейс push. Push имеет параметр типа Par и возвращает результат типа Res (представим код для Java-подобного языка).
B:

```
Res push(Par);
```

Рассмотрим различные производные (от push) вызовы, основанные на том, что модуль B масштабируем.

Объявление функции	**Описание работы**
`Res push_any(Par);`	Вызвать метод на произвольном экземпляре B
`ArrayList<Res> push_all(Par);`	Вызвать метод на всех экземплярах B
`ArrayList<Res> push_all_with_filter(Par, Filter);`	Вызвать метод на всех экземплярах B с последующей фильтрацией результатов
`ArrayList<Res> push_parallel(ArrayList<Par>);`	Выполнить N вызовов push одновременно на N экземплярах B (подобно функции map с модификатором push)

Все описанные выше вызовы синхронные, но существуют и их асинхронные версии.

функционального кода, использующего такие вызовы, приведен в таблице.

Прозрачность и простота представленной модели позволяют легко описывать с её помощью сложные системы, но при этом она полностью отстраняется от таких обязательных моментов реализации, как механизм обнаружения, именования, распределения (планирования) ресурсов и коммуникации (межмодульные вызовы производятся как

обычные локальные). Следовательно, для того, чтобы построить распределенную систему на основе этой модели, необходима некоторая система исполнения, которая обеспечила бы все эти механизмы.

Далее изложена архитектура системы исполнения и каркаса программной системы, реализующие работу с описанной выше моделью.

Главной идеей, положенной в основу предлагаемой реализации, является предоставление программисту возможности создавать эффективную распределенную систему, работая лишь в терминах абстракции модели и явно кодируя только её функциональный код.

Каркас программной системы состоит из двух основных частей:

- служебной части, отвечающей за связь с системой исполнения;
- шаблонов модулей.

Мы предлагаем реализацию, в которой программный каркас будет генерироваться на основе спецификаций модулей системы. Такая спецификация представляет собой:

1. Описание всех составляющих модулей системы (поведения, вызовы) за исключением функционального кода.
2. Служебную информацию о языке программирования, на котором планируется реализация отдельных модулей системы.
3. Описание объектов передачи данных, которые являются аргументами в межмодульных вызовах.

Используя эту информацию, программа-генератор кода выполнит построение каркаса приложения. После этого на основе полученных шаблонов модулей программисту потребуется лишь закодировать функциональную часть программы. Схема описанного жизненного цикла представлена на рисунке 2.

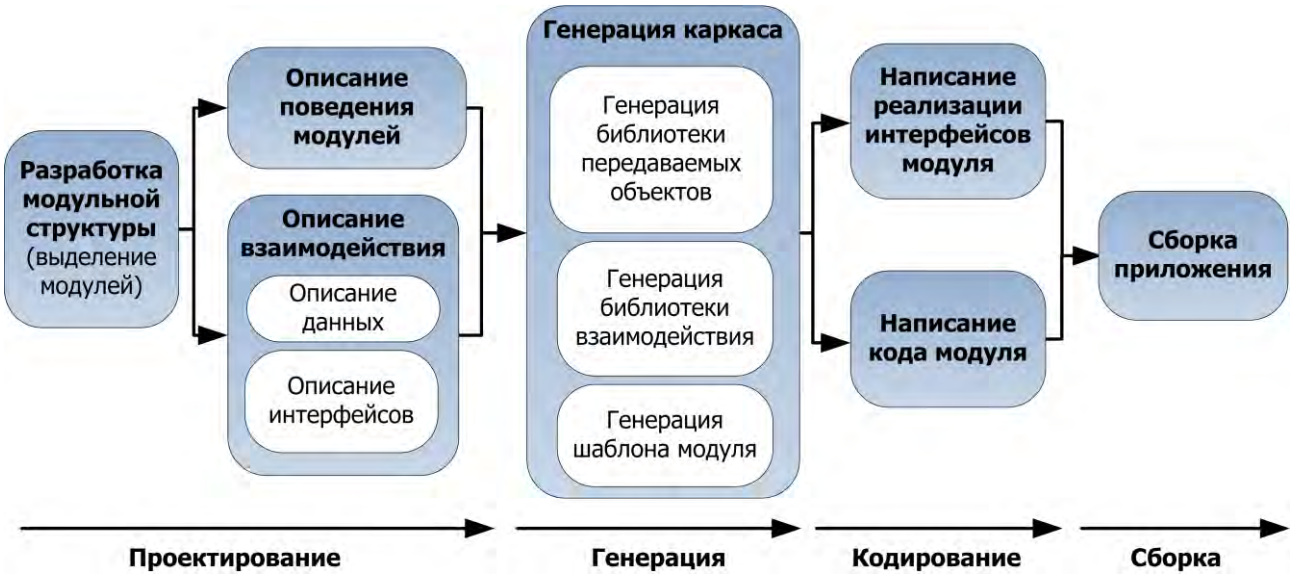


Рисунок 2.. Жизненный цикл программной системы

Полученная таким образом программа может эксплуатироваться в специально организованной системе исполнения. Система исполнения (см. рисунок 3) представляет собой программный комплекс, обеспечивающий использование ресурсов вычислительной среды, адресацию, поддержание поведения программы (модулей, вызовов, программной системы в целом), службу именования.

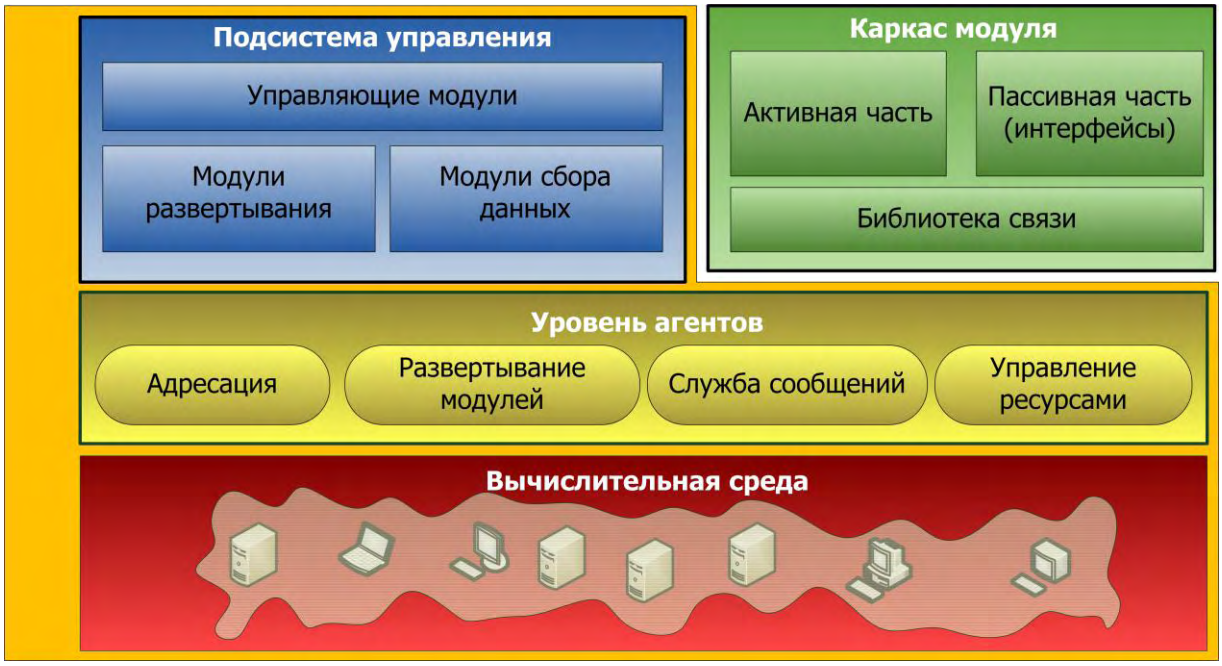


Рисунок 3. Система исполнения

Вычислительной средой для программ, написанных на основе предложенной технологии, в общем случае, является гетерогенный комплекс, базовые аппаратные компоненты которого представляют собой компьютерные ресурсы организации, объединенные средствами локальной сети.

Подключение к системе отдельного узла осуществляется путем установки и настройки программы-агента. Настройка включает в себя предварительный анализ системно-аппаратной части компьютера, установку квот на ресурсы, потребляемые агентом и приложениями.

Для контроля среды исполнения используется подсистема управления. Она берет на себя функции:

- сбора служебных данных о состоянии агентов, текущей нагрузке (на каналы связи, аппаратуру), сбоях;
- агрегации и анализа полученных данных;
- преобразование системы к более эффективному виду (за счет запуска, остановки и переноса экземпляров модулей).

Стратегия преобразования системы может быть выбрана из нескольких вариантов, например:

1. «Компактная система» Система растет постепенно из одного узла. По мере роста нагрузки на отдельные ее части, она начинает их переносить и дублировать, в итоге принимает стабильное состояние.
2. «Жадная система» Система стартует все части на разных узлах, тем самым занимая большую часть свободных ресурсов.
3. «Упреждение пиков нагрузки» Производится за счет предварительного резервирования нагруженных модулей.

Запуск программы производится с помощью интерфейсной части подсистемы управления. В ней среда исполнения представляется пользователю в виде взвешенного графа, в котором вес ребра соответствует скорости передачи данных между узлами, а вес вершины - её относительной производительности. Мониторинг состояния среды можно вести в динамике, отслеживая текущую нагрузку на узлы и каналы связи.

Для запуска программы необходимо выбрать исполняющий подграф, удовлетворяющий конфигурации программы (в частном случае он может состоять из одной вершины) и дать команду на развертывание приложения.

Приложение может быть запущено в одном из следующих режимов:

- режим разработки – предоставляет наиболее полную отладочную информацию (сведения о межмодульных вызовах, состоянии очередей сообщений...);
- режим тестирования на нагрузку, как всей системы исполнения, так и отдельного модуля;
- режим эксплуатации – система работает в режиме максимальной производительности, при этом доступна только обобщенная информация о нагрузке на узлах. Масштабирование осуществляется автоматически согласно выбранной стратегии.

Предложенная архитектура системы исполнения позволит не только провести всестороннее исследование эффективности и удобства разработанной нами модели, но и сама является сложной многоуровневой распределенной системой. От качества её реализации напрямую зависят такие ключевые параметры создаваемых приложений, как быстродействие, надежность, масштабируемость. Поэтому мы считаем целесообразным в ходе дальнейших изысканий обратить внимание на совершенствование ее компонентов: алгоритмов принятия решений, используемых управляющей подсистемой (внедрение алгоритмов, базирующихся на идее самообучающихся систем); библиотеки инструментальных средств; подсистемы управления (применение механизма голосования и выборов для организации устойчивости ее архитектуры).

Список литературы

1. Таненбаум Э., Ван Стеен М. Распределенные системы. Принципы и парадигмы . — СПб.: Питер, 2003. — 877 с.
2. Joe Armstrong. Programming Erlang. Software for a Concurrent World. - The pragmatic bookshelf. 2007.
3. Michi Henning. A New Approach to Object-Oriented Middleware // IEEE Internet Computing, vol. 8, no. 1, pp. 66-75, Jan./Feb. 2004.
4. Henning, M. Massively Multiplayer Middleware. Queue 1, 10 (Feb. 2004), 38-45.
5. Weiss, A. Computing in the clouds. netWorker 11, 4 (Dec. 2007), 16-25.

РАЗРАБОТКА ПОЛНОТЕКСТОВОЙ ЭЛЕКТРОННОЙ БИБЛИОТЕКИ РЕДКИХ ПЕРИОДИЧЕСКИХ ИЗДАНИЙ АКУНБ ИМ. В.Я ШИШКОВА

Кожевин И.И. – студент, Лукоянычев В.Г. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

В настоящее время в России ведется создание и отработка законодательной базы в области авторского права и смежных прав. На библиотеки, предоставляющие доступ к изданиям, защищенным авторским правом, в электронном виде теперь наложены достаточно жесткие правила, имеющие своей целью выполнение не так давно введенных законодательных норм. Однако при этом достаточно сильно страдают сами читатели, получать необходимую информацию из документов становится достаточно неудобно, и на второе место после соблюдения законодательства выходит удобство читателей. Библиотеки ищут решение данной проблемы, и одним из выходов является создание программного

комплекса, позволяющего читателям получать доступ к библиотеке к удобной для них форме, а библиотеке не беспокоиться о возможных нарушениях законодательства.

Требовалось разработать электронную библиотеку для просмотра редких периодических изданий фонда АКУНБ им. И.И. Шишкова. Она должна была предоставлять возможности поиска документа по ключевым словам, просмотра списка документов по разделам, постраничного просмотра выбранного документа.

Электронная библиотека представляет собой сайт, состоящий из трех частей:

- административной, через которую администратор сможет добавлять новые и изменять уже добавленные документы, управлять деревом каталогов документов, управлять списком пользователей и администраторов;
- пользовательской, через которую пользователь может осуществлять просмотр дерева каталогов, просмотри информации о документах и поиск документов;
- модуля простотра документов, представляющего собой java-сервлет, которых по HTTP-запросу будет постранично выдавать выбранный пользователем документ.

Модуль просмотра документов производит добавление ссылок на предыдущую и следующую страницы, реализует возможность для перехода на любую страницу документа, нанесение водяных знаков на документы, производит конвертацию страницы документа в картинку для защиты от копирования текста и шифрование выдаваемой страницы с защитой от печати, изменения и сохранения. Модуль не кеширует созданные страницы на сервере для экономии места, так как PDF-документы занимают достаточно большой объем.

ПРИМЕНЕНИЕ МЕТОДА МОНТЕ-КАРЛО ПРИ РЕШЕНИИ ЗАДАЧИ КЛАСТЕРИЗАЦИИ

Колосовский М.А. – студент, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В общем случае задача ***кластеризации*** заключается в разбиении множества некоторых объектов на классы по некоторому признаку, например, этим признаком может стать минимальность расстояния между объектами одного класса (расстояние, в свою очередь, может также задавать множеством способов). Например, пусть нам дан большой набор цветов в формате RGB, используемых в изображении, требуется заменить этот набор более компактным, предполагается выделить группы цветов (в каждой группе будут только похожие цвета) и из каждой группы взять по одному представителю.

В данной статье будет рассматриваться двумерная геометрическая задача выделения «скоплений точек» на плоскости. Для простоты рассмотрения мы будем считать ***скоплением*** окружность с включенными в нее точками. ***Плотностью скопления*** будем считать отношение числа точек внутри и на границе к окружности к площади окружности. Итак, все множество точек нужно разбить на непересекающийся по точкам набор скоплений.

Метод Монте-Карло – это вероятностный метод для решения самых разнообразных задач, независимо от того из какой области эта задача, будь это вычислительная математика, вычислительная геометрия, теория графов, искусственный интеллект. Суть метода Монте-Карло заключается в проведении некоторого количества независимых случайных испытаний, после чего искомая величина выводится статистическим путем, например, искомой величиной признается среднее арифметическое величин, полученных в проведенных испытаниях. Таким образом, результаты, вычисленные этим методом, носят вероятностный характер, т.е. теоретически могут отличаться от правильных на сколько угодно большую

величину. Однако вероятность больших отклонений крайне мала. Точность метода зависит от количества испытаний: чем больше испытаний, тем точность выше. Метод прост для понимания и в реализации, хотя и не дает точного решения, как аналитические методы. В ситуациях, когда невозможно аналитически вывести решение, метод Монте-Карло может прийти на помощь. В задачах искусственного интеллекта (ИИ), как правило, не требуется точного решения, поэтому этот метод может найти применение и в других задачах ИИ.

Решая задачу кластеризации методом Монте-Карло, бросим на нашу плоскость с точками множество центров наших скоплений. Для каждого центра отсортируем все кластеризуемые точки по возрастанию евклидова расстояния до этого центра. Пробежимся по списку для данного центра, i-ая (нумерация с 1) точка находится на расстоянии r_i, тогда получаем вокруг данного центра скопление с плотностью $i/(\pi r_i^2)$. Здесь следует отметить, что нужно ввести дополнительные ограничения на скопления, например, ввести минимальный радиус скопления, в противном случае, алгоритм будет строить скопления с нулевым радиусом. Таким образом, бросив M центров, кластеризуя N точек, мы получим NM скоплений с вычисленными плотностями.

Полученные скопления пока пересекаются по точкам, теперь требуется выделить из них набор скоплений, которые по точкам не пересекаются. На данном этапе существует несколько вариантов работы алгоритма, но мы воспользуемся следующим. Отсортируем все скопления по убыванию плотности. Будем постепенно набирать скопления в результирующий набор. Точки, которые входят в одно из скоплений из результирующего набора, будем называть ***занятыми***, остальные ***свободными***, сначала все точки свободны. Пойдем с самых «плотных» скоплений (т.е. с начала списка). Если очередное скопление содержит занятые точки, то пропустим его, если только свободные, то добавляем его в результирующий набор. Таким образом, мы получим набор скоплений непересекающихся по точкам.

Выше описан самый простой вариант алгоритма с применением метода Монте-Карло, алгоритм можно улучшить, введя некоторые дополнительные элементы. Для случайно выбранного центра можно рассматривать не все N точек, а лишь те, которые наиболее близко расположены к нему. Когда для данного скопления проверяем, какие точки в него входят (свободные или занятые), можно избежать проверки всех N точек, а проверять лишь те, которые входят в данное скопление.

Практическим применением данной задачи и данного алгоритма ее решения может быть, например, задача о размещении башен связи. Каждая башня снабжает сигналом определенную площадь, охватываемую кругом данного радиуса. Радиус этого круга зависит от мощности станции, более мощные станции стоит дороже. Таким образом, будет выгодно поставить башни в центре больших скоплений абонентов, например, внутри крупного каждого населенного пункта, как впрочем, и делается в действительности.

Данный алгоритм применим не только к рассмотренной двумерной задаче, метод Монте-Карло можно использовать для решения абсолютно любой задачи кластеризации. Например, он применим для решения рассмотренной выше задаче об оптимизации (сжатии) палитры цветов. Можно случайно бросать центры шаров в трехмерном пространстве цветов RGB. Эти шары будут представлять близко расположенные цвета, а их центры будут соответствовать цветам, которые будут присутствовать в оптимизированной палитре.

Таким образом, вероятностный метод Монте-Карло является крайне мощным средством для решения всевозможных задач из области искусственного интеллекта, для которых невозможно найти простого точно решения и его заменяют приближенным легко вычислимым решением.

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ АНАЛИЗА КАЧЕСТВА САЙТОВ НА ОСНОВЕ МЕТОДОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Константинов А.Ю. – студент, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Каждый день в Интернете появляются новые сайты. Одни из них являются профессионально выполненным продуктом с высокой функциональностью и приятным и эргономичным дизайном. Другие представляют собой любительские странички с отсутствием полезной информации и с отталкивающим дизайном. Пользователям, безусловно, хочется работать с первым типом сайтов. Поэтому актуальной является проблема автоматической оценки качества сайтов. Автоматизированная система оценки качества сайтов может быть, например, интегрирована в поисковые системы с целью сортировки сайтов одного содержания в порядке убывания их качества.

На сегодняшний день в мире существует достаточно мало аналогов автоматической оценки качества сайтов, поскольку это малоизученный вопрос. Поэтому было принято решение создать собственный программный продукт, функционирующий по алгоритмам, разрабатываемым автором в ходе научной работы.

В предлагаемой работе оценка качества сайтов основана на работе нейросети. В роли критериев качества сайтов автором выбраны два параметра: цветовая палитра и компоновка элементов дизайна.

В настоящее время ведётся работа по реализации оценки сайта по цветовой палитре, поэтому в данной работе подробно описана оценка качества сайта именно по этому критерию.

Из теории цвета известно, что одни комбинации цветов приятнее для глаза человека, чем другие. На этом факте основывается проект Владимира Голованова и Андрея Якушева «Набор палитр для Web-дизайна». Создатели проекта собрали в одном источнике более четырёхсот палитр (состоящих, в основном, из шести цветов), которые помогут Web-разработчикам ориентироваться в цветовом решении создаваемого сайта. Обучающая выборка для нейросети составлена из семидесяти палитр, входящих в «Набор палитр для Web-дизайна» (по десять из каждой тональности, представленной в наборе), и семидесяти палитр, сгенерированных программой «Palettes Generator», созданной автором. Данная программа генерирует палитры, которые сохраняются по желанию пользователя в XML-файл. В данном случае сохранялись палитры с несочетающимися цветами. Палитрам из «Набора палитр для Web-дизайна» присвоена метка «хорошая палитра», палитрам, сгенерированным программой, присвоена метка «плохая палитра». Обучающая выборка хранится в XML-файле специального формата.

Оценка качества сайта основана на работе нейросети. Нейросеть имеет 18 входов, 1 выход, 5 слоёв (включая входной и выходной). В первом слое (входном) расположены 18 нейронов с одним входом; во втором слое расположены 36 нейронов с 18-ю входами; в третьем слое расположены 18 нейронов с 36-ю входами; в четвёртом слое расположены 9 нейронов с 18-ю входами; в пятом слое (выходном) расположен 1 нейрон с 9-ю входами. Вид нейросети, созданный с помощью разработанного автором конструктора нейросетей «Designer», представлен на рисунке 1. Нейроны двух соседних слоёв соединены друг с другом по принципу «каждый с каждым». В качестве активационной функции нейронов используется логистическая сигмоидальная функция с коэффициентом кривизны 0,1.

Обучение нейросети производится одним из пяти алгоритмов, реализованных в конструкторе нейросетей. Самый простой из них – метод обратного распространения ошибки – даёт приемлемый результат: 2 240 эпох.

Палитры в обучающей выборке состоят из шести цветов. Каждый цвет кодируется тремя составляющими: красная, зелёная, синяя. В итоге получается 18 числовых значений, которые подаются на 18 входов нейросети (одно числовое значение на один вход). После завершения

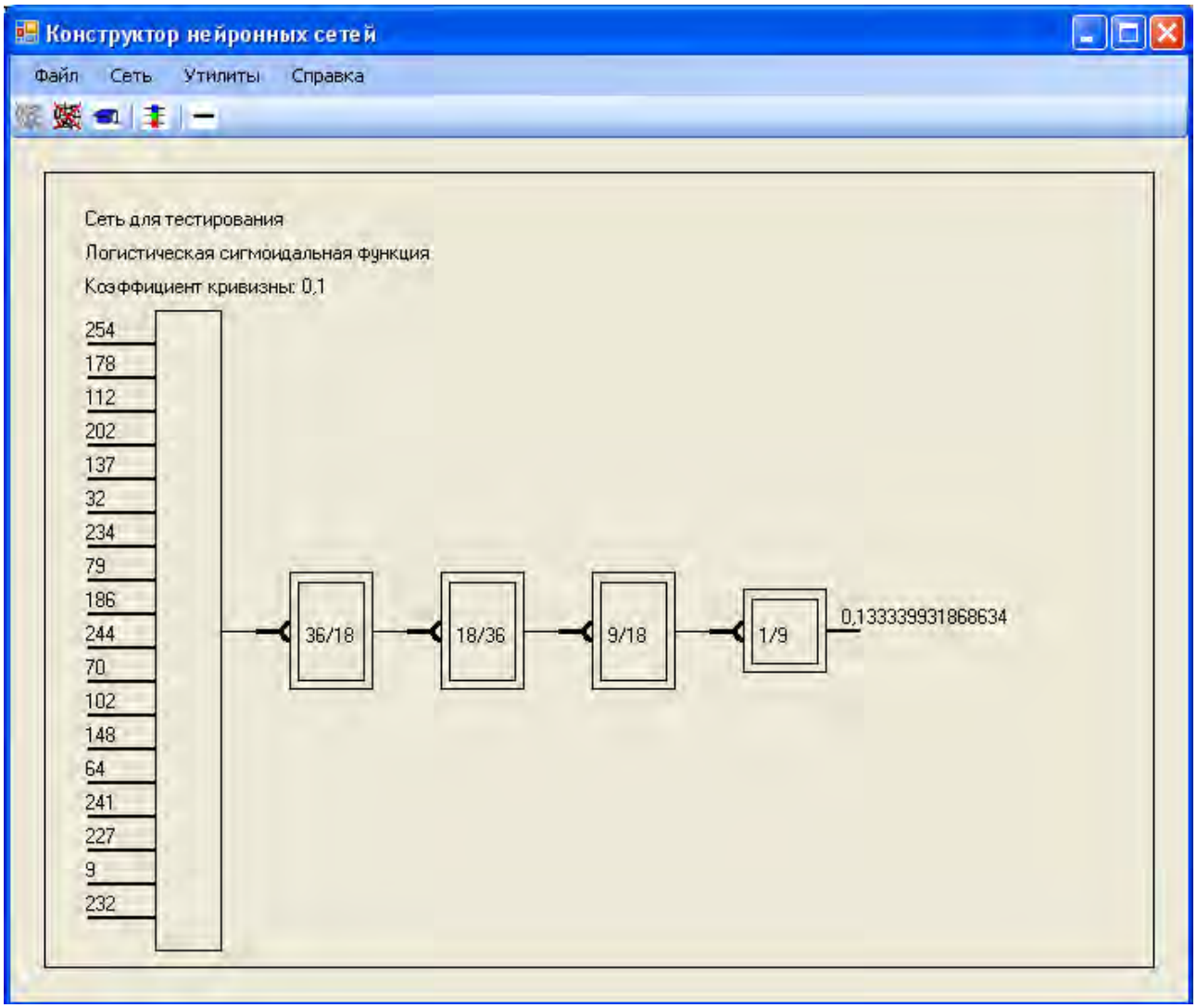


Рисунок 1 – Нейросеть для оценки качества сайта по цветовой палитре

работы нейросети, единственный нейрон в выходном слое имеет выходное значение в интервале (0; 1). Алгоритм обучения корректирует веса нейросети так, чтобы «хорошим» палитрам соответствовал выход нейросети, значение которого стремится к 1, а «плохим» палитрам соответствовал выход, значение которого стремится к 0. График обучения нейросети представлен на рисунке 2 (на графике видны только 500 эпох). На оси абсцисс отображаются эпохи обучения нейросети, на оси ординат отображается суммарная ошибка обучения по всем палитрам из обучающей выборки на каждой эпохе. Обучение прекращается в том случае, если ни одна палитра не даёт ошибку обучения, большую допустимой величины ошибки распознавания одной палитры.

После обучения нейросети система готова к проведению серии экспериментов. В системе предусмотрены два режима: оценка палитры сайта и оценка палитры изображения. Оценка палитры сайта отличается от оценки палитры изображения единственным промежуточным этапом: система оценки с помощью специального алгоритма делает «снимок» сайта, открытого в Интернет-браузере «Internet Explorer» (то есть переводит сайт в изображение). Алгоритм оценки работает следующим образом: с помощью алгоритма

кластеризации палитры из изображения выделяются шесть основных цветов; затем каждый цвет разбивается на три компоненты, и данные значения подаются на входы обученной нейросети; после этого запускается вычислительный процесс в нейросети; в завершении вычислительного процесса система оценки выдаёт значение нейрона, расположенного в выходном слое. Будем считать, что сайты или изображения, которым соответствует значение

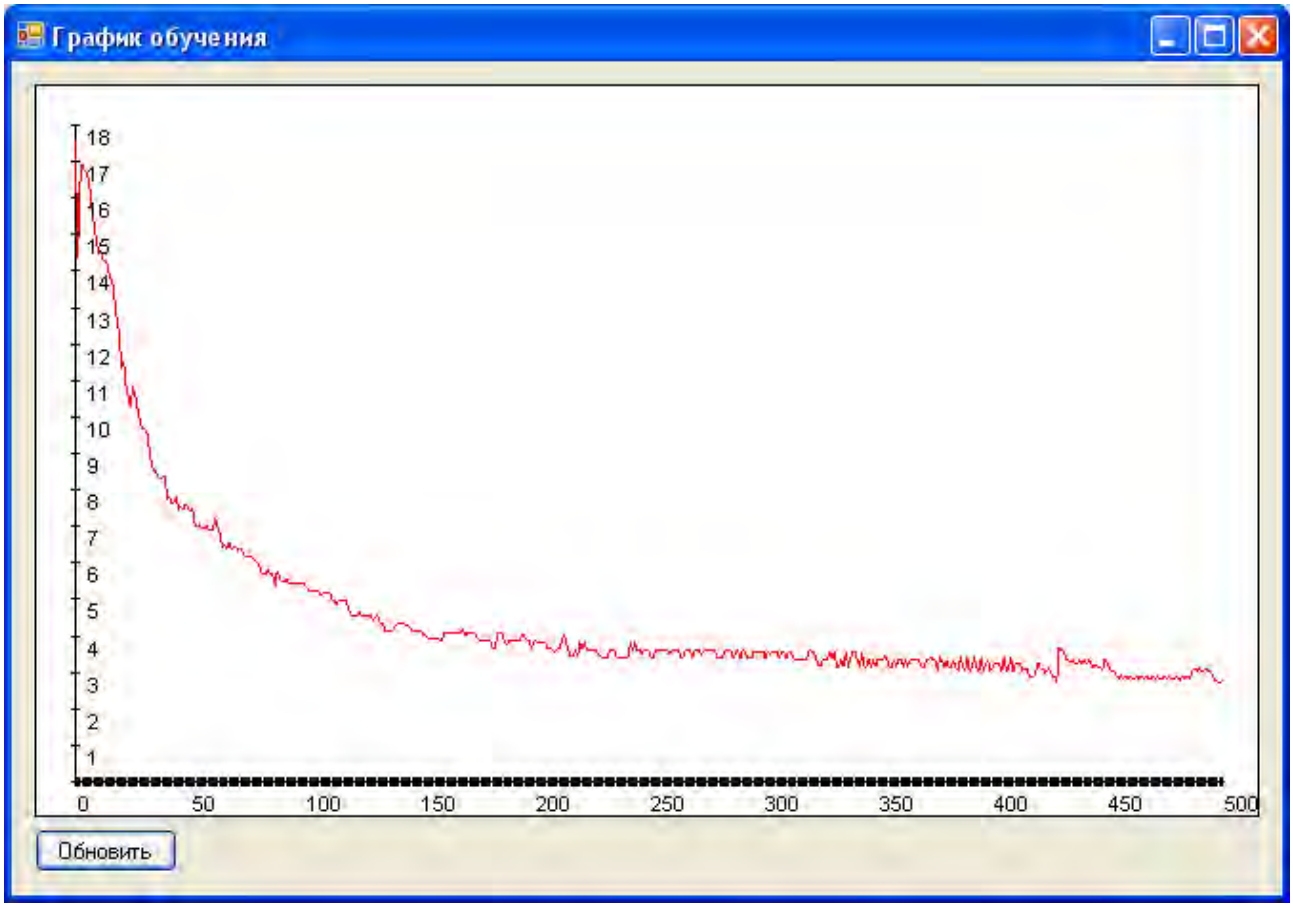


Рисунок 2 – График обучения нейросети методом обратного распространения ошибки

выходного нейрона, большее 0,5, имеют палитры с меткой «хорошая», а сайты или изображения, которым соответствует значение выходного нейрона, меньшее 0,5, имеют палитры с меткой «плохая».

Автором был проведён ряд экспериментов, заключающихся в оценке качества палитр различных сайтов и изображений. Эксперименты позволили сделать вывод о работе программного продукта. Система ставит более высокую оценку сайтам и изображениям с палитрой, в которой цвета расположены ближе друг к другу, чем сайтам и изображениям с пёстрой палитрой. Этот факт можно объяснить тем, что в обучающую выборку попали в основном «приглушённые» палитры из «Набора палитр для Web-дизайна». Система оценки с вышеописанным свойством имеет ценное практическое значение: представляется возможным выделять из множества сайтов те, которые не будут раздражать пользователя своей пестротой; также с помощью данной системы можно распознавать сайты с обилием рекламных баннеров, которые своими яркими красками будут снижать оценку сайта.

Автором планируется внести несколько усовершенствований в систему: создать несколько вариантов обучающей выборки для того, чтобы высокая оценка выставлялась не только сайтам с «приглушённой» палитрой, но и сайтам с более яркими, но всё же приятными человеческому глазу палитрами. Планируется внести усовершенствования в

алгоритм выделения палитры из изображения (например, можно игнорировать фоновый цвет, оценивая только те цвета, с помощью которых отображается контент - содержимое сайта). Также планируется разработать и реализовать алгоритм оценки компоновки элементов дизайна, что позволит оценивать интерфейс сайта более полно и многопланово.

МЕХАНИЗМЫ ПОПОЛНЕНИЯ ЗНАНИЙ МОДЕЛИ ЛОГИЧЕСКОГО ВЫВОДА НА ОСНОВЕ ЛЕКСИКОНА

Крайванова В.А. – аспирант, Крючкова Е.Н. – к.ф-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Современные инструменты человеко-машинного взаимодействия строги и формальны, они требуют от пользователя определенных знаний и опыта работы. В данной работе рассматривается модель логического анализа фраз на естественном языке, позволяющая осуществлять перевод синтаксической конструкции с естественного языка на формальный язык команд объекта управления путем замены фрагментов синтаксических конструкций на их логические следствия.

Пусть W – конечное множество известных в модели слов. Нечеткое отношение обобщения $Gen: W \times W \rightarrow [0..1]$ определяет степень уверенности модели в том, что некоторое понятие $p \in W$ обобщает слово $w \in W$. Нечеткое отношение синонимичности $Syn: W \times W \rightarrow [0..1]$ определяет степень уверенности системы в том, что некоторое слово $w_A \in W$ является синонимом к слову $w_B \in W$. В качестве представления фраз на ЕЯ нами выбрано дерево, узлы которого помечены словами из лексикона W, а ребра – отношениями между ними[1]. Такое помеченное дерево фразы (ПДФ) в том или ином виде строят практически все синтаксические анализаторы. Задача логического анализа фразы заключается в том, чтобы свести некоторую начальную фразу ϕ к множеству фраз, определяющих команды и понятных объекту управления. Механизм преобразования фраз описывается конечным множеством правил контекстной замены ***Rules.*** В общем виде правило задается парой $r = <H \rightarrow C, Parameters>$, где H и C – гипотеза и следствие правила, представленные двумя ПДФ, некоторые вершины которых, возможно, помечены не словами из W, а параметрами. ***Parameters*** – множество параметров правила. Значением параметра p_i может стать ПДФ, корень которого помечен словом, обобщаемым некоторым множеством понятий N_i из множества всех понятий предметной области. Команда объекта управления представляет собой заключительное правило, для которого C – пустое дерево. Операция унификации $<\gamma, Values>=Unification(\phi, r)$ - это операция подстановки дерева ϕ в правило r с учетом параметров. ***Values*** – значения параметров, γ - уровень релевантности результата.

Важным свойством интеллектуальных моделей является обучаемость, способность сохранять свою адекватность в изменяющихся внешних условиях. Лексикон модели является более простым и прозрачным уровнем знаний модели. Рассмотрим механизм пополнения лексических знаний в процессе функционирования.

Достраивание отношений ***Gen*** и ***Syn*** основано на анализе позиции в дереве фразы, на которой встретилось незнакомое слово. Введем обозначения. $AssGen_0$ - начальный уровень достоверности предположения о существовании отношения обобщения между словами. ***AssGen*** - уровень подтверждения достоверности предположения о существовании отношения обобщения. $AssSyn_0$ - начальный уровень достоверности предположения о существовании отношения синонимичности между словами. ***AssSyn*** - уровень подтверждения достоверности предположения о существовании отношения синонимичности.

Чтобы модифицировать отношение ***Gen***, модель строит предположение о том, что слово, подставляемое в параметр, соответствует данному параметру. Пусть $<\gamma, Values>=Unification(\phi, r)$ – результат унификации гипотезы некоторого правила r и дерева

ϕ, и $\exists p_i$: $\tau(p_i) < 1$, где $\tau(p_i)$ - уровень достоверности унификации параметра. Тогда, если $\gamma > \gamma_0$, то отношение Gen изменяется следующим образом:

$$Gen_{old} \rightarrow Gen_{new}:$$

$$Gen_{new}(w,n) = Gen_{old}(w,n), w \neq w_i \text{ или } n \notin N_i;$$

$$Gen_{new}(w_i,n) = \begin{cases} AssGen_0, & Gen_{old}(w_i,n) = 0 \\ \min(Gen_{old}(w_i,n) + AssGen, 1), & Gen_{old}(w_i,n) > 0 \end{cases}, n \in N_i$$

где w_i – слово, которым помечен корень значения параметра p_i.

Отношение синонимичности достраивается на основе предположения, что два унифицируемых узла помечены синонимами. Рассмотрим унификацию узлов $u \in \phi$ и $u_r \in C$. Здесь ϕ - анализируемая фраза, C - гипотеза правила r. Введем обозначения: w – слово, которым помечен узел u, w_r – слово, которым помечен узел u_r, $<\gamma, Values> = Unification(\phi, r)$.

Тогда, если $\gamma > \gamma_0$, то отношение Syn изменяется следующим образом:

$$Syn_{old} \rightarrow Syn_{new}:$$

$$Syn_{new}(w_1,w_2) = Syn_{old}(w_1,w_2), w_1 \neq w \text{ или } w_2 \neq w_r;$$

$$Syn_{new}(w,w_r) = \begin{cases} AssSyn_0, & Syn_{old}(w,w_r) = 0 \\ \min(Syn_{old}(w,w_r) + AssSyn, 1), & Syn_{old}(w,w_r) > 0 \end{cases}$$

Уверенность модели в сделанном предположении накапливается в процессе логического анализа. Если из результата ϕ_{new} применения правила r к функциональной форме ϕ не было получено ни одной заключительной функциональной формы, то все предположения, сделанные при формировании ϕ_{new} считается неподтверждённым и лексикон возвращается в состояние Gen_{old} и Syn_{old}.

Рассмотрим возможности модификации множества правил на основе изменений, внесенных в лексикон. Логические знания близких предметных областей имеют схожее внутреннее строение. Правила имеют ту же структуру, отличаясь лишь на лексическом уровне. Тема $T \subseteq W$ - это подмножество слов лексикона, принадлежащих к одной предметной области. В отличие от отношения обобщения, которое определяет для данного слова понятие на более высоком уровне абстракции, принадлежность к теме фиксирует раздел предметной области, к которой относится слово.

Пусть T и G - некоторые темы. На множествах слов $A_T \subseteq T$ и $A_G \subseteq G$ установлено отношение аналогии $A_T \sim A_G$, если:

- между элементами множеств A_T и A_G установлено взаимно однозначное соответствие;
- при замене в произвольной фразе ϕ всех слов $t \in A_T$ на соответствующие им слова $g \in A_G$ получаемая при этом фраза ϕ' остается осмысленной при условии, что $\forall w \in T: \exists u \in \phi: Word(u) = w \Rightarrow w \in A_T$.

Обозначим $t = Association(g)$. Отношением аналогии могут быть связаны только слова, отнесённые к некоторым темам.

Состояние знаний модели описывается парой $<W, Rules>$, где W - лексикон модели, $Rules$ - множество правил модели. Пусть $<W_0, Rules_0>$ - начальное состояние знаний модели. Рассмотрим изменение множества правил при добавлении в лексикон новой темы T_N. Введем обозначения. $A_N \subseteq T_N$ - множество слов и понятий новой темы, связанные отношением ассоциации с другими словами. $AssThemes(A_N) = \{T | \exists A \in T: A \sim A_N\}$ - множество тем, со словами из которых множество A_N связано отношениями ассоциации. $Nodes_A(r)$ - множество узлов правила r, помеченных словами из темы T. Тогда $<W_{new}, Rules_{new}>$ - новое состояние знаний модели, где $W_{new} = W_0 \cup T_N$, $Rules_{new} = Rules_0 \cup Rules_A$). $Rules_A$ - множество всех новых правил $r_A = <H_A \rightarrow C_A, Parameters_A>$. Эти правила получены из уже имевшихся правил $r = <H \rightarrow C, Parameters> \in Rules_0$, таких, что $\exists T \in AssThemes(A_N): Nodes_A(r) \neq \varnothing$ и $Nodes_A(r) = Nodes_T(r)$. Преобразование осуществляется следующим образом:

- Деревья H_A и C_A получены из деревьев H и C соответственно заменой пометок узлов u $\in Nodes_A(r)$ на ассоциированные с ними слова $Association(Word(u))$ из темы T_N.
- Параметр $p_{Ai}=p_i$, $p_{Ai} \in Parameters_A$, $p_i \in Parameters$, если $p_i=<name, N>$, $\forall w \in N: w \notin A$.
- Параметр $p_{Ai}=<name, N_A>$, $p_{Ai} \in Parameters_A$, $p_i=<name, N> \in Parameters$, $N_A=N \setminus A \cup \{w_A | w_A=Association(w), w \in N \cap A\}$, если $\exists w \in N: w \in A$.

Построенный механизм позволяет автоматизировать рутинные операции пополнения знаний в пределах одной предметной области.

Предложенная математическая модель реализована в архитектуре AL System. Вычислительный эксперимент показал адекватность модели.

Список литературы

1. Крайванова В.А. Методы организации базы знаний системы логического анализа текстов на естественном языке - Перспективные технологии искусственного интеллекта: сборник трудов научно-практической конференции (г. Пенза, 1-6 июля 2008 г.). - Пенза: Информационно-издательский центр ПензГУ, 2008. - стр. 202-207

АНАЛИЗ СТРУКТУРЫ, СОСТАВА И ГРАНУЛОМЕТРИЧЕСКИХ ХАРАКТЕРИСТИК ЗЕРНА

Кузюков С.А., Корней В.И., Семейкин А.Г. – студенты
Лузев В.С. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

В производстве возделывающем и перерабатывающем зернопродукты, существует ряд проблем, связанных с анализом качественных характеристик зерна. Проблемы возникают в связи с двумя причинами:

1) зёрен ОЧЕНЬ МНОГО, анализировать необходимо если не все, то значительную их часть.
2) анализ выполняет человек со всеми вытекающими отсюда последствиями.

Представленная научно-исследовательская работа посвящена решению задачи автоматизации такого анализа, а именно:

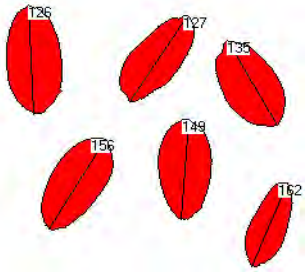
- увеличение скорости и точности анализа
- уменьшение участия при этом человека

В работе предполагается исследовать и автоматизировать 3 вида анализа зерна.

1. Анализ мучнистости/стекловидности зёрен пшеницы

Этот анализ предназначен для определения качества зерна. Процент мучнистости/стекловидности зёрен имеет большое значение для производителей хлебобулочных изделий, особое внимание этому показателю уделяют производители макарон. Данный анализ осуществляется следующим образом: оператор или машина разрезает пополам зёрна пшеницы, затем визуально анализируются срезы: по ним видно, какую структуру имеет зерно: мучнистую или стекловидную. Больше ценятся зёрна, имеющие стекловидную структуру.

2. Гранулометрический анализ зернопродуктов



Этот анализ предназначен для определения гранулометрических характеристик зёрен, то есть их размеров: длины периметра зёрнышка и его объёма (либо площадь проекции). Применяется анализ для различных зернопродуктов. Применяется прежде всего в сельском хозяйстве, а так же в промышленности, где имеет значение информация о размерах зёрен и о том, насколько размеры различных зёрен различаются.

3. Анализ засорённости зёрен

Этот анализ предназначен для определения того, насколько чистым/сорным является зерно, то есть несколько инородных частиц содержится в зерне. От предметов, имеющих заметно другие размеры, зёрна отделяются автоматически, проходя стадии конвейера, куда сложнее отличить зёрна возделываемых растений от предметов, близких по размеру, особенно от семян сорняков. Анализ применяется как в сельском хозяйстве, так и на промышленных предприятиях: производство мучных изделий, пивоварение.

В данной научно-исследовательской работе предполагается ввод изображений с быстродействующей цифровой камеры, которая делает снимки зёрен, поступающих с конвейера.

Полученные цифровые изображения отправляются на компьютер, который производит анализ этих изображений с помощью комплекса программ. Конечная цель работы заключается в разработке программного обеспечения, реализующего изложенный выше анализ.

Существуют некоторые программные решения данных задач, но они имеют ряд существенных недостатков, в частности они имеют недостаточно точную степень анализа и ограниченную функциональность.

Для достижения цели предполагается использование алгоритмов распознавания образов и методов «искусственного интеллекта», таких, как нейронные сети.

Список литературы

1. Хайкин Саймон Нейронные сети: полный курс, 2-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2006. – 1104 с.

ВИРТУАЛЬНАЯ ЛАБОРАТОРИЯ ПО ТЕОРИИ УПРАВЛЕНИЯ

Кулик А.Г – студент
Алтайский государственный технический университет (г. Барнаул)

Виртуальные лаборатории – обучающие системы, моделирующие поведение объектов реального мира в компьютерной среде. Они помогают учащимся овладевать новыми знаниями и умениями в естественнонаучных дисциплинах, таких как химия, физика, биология, прикладная математика. Суть виртуальных лабораторий можно отразить тремя ключевыми словами: эксперимент, моделирование, динамика.

Данная программная система представляет собой виртуальную лабораторию по теории управления. В ней решается задача синтеза оптимальной по критерию H_2 системы

управления на примере активного управления подвеской транспортного средства и задача синтеза оптимальной по критерию H_∞ системы управления на примере управления продольным движением самолета.

В первом случае объектом управления является механическая система, состоящая из двух масс (колеса и корпуса транспортного средства), соединенных пружиной и демпфером. Динамика системы в отклонениях от положения равновесия описывается уравнениями:

$$m_1\frac{d^2h_1}{dt^2} = -k_1(h_1 - h_0) + k_2(h_2 - h_1) + b\left(\frac{dh_2}{dt} - \frac{dh_1}{dt}\right) - F,$$
$$m_2\frac{d^2h_2}{dt^2} = -k_2(h_2 - h_1) - b\left(\frac{dh_2}{dt} - \frac{dh_1}{dt}\right) + F,$$

где h_1- вертикальное перемещение колеса, h_2 - вертикальное перемещение корпуса, F - управляющая сила.

Во втором случае объектом управления является самолет, продольное движение которого в отклонениях от заданной траектории полета описывается уравнением:

$$\dot{x} = Ax + Gu + Sw,$$

где x - вектор состояния, u - управление, w - возмущение. Вектор состояния включает в себя отклонение угла наклона траектории, отклонение угла тангажа, отклонение угловой скорости тангажа и отклонение высоты.

Критерии H_2, H_∞ представляют собой нормы передаточной матрицы замкнутой системы [1].

$$H_2 = \left(\frac{1}{2\pi}\int_{-\infty}^{\infty} tr\ \hat{G}^T(-j\omega)\hat{G}(j\omega)d\omega\right)^{\frac{1}{2}},$$
$$H_\infty = \sup\sigma(\hat{G}(j\omega)),$$

где $\hat{G}(j\omega)$ – передаточная матрица, $\sigma(\cdot)$ – максимальное сингулярное число матрицы.

В разрабатываемой лаборатории предусмотрено детальное описание каждого из этих экспериментов, вывод соответствующих графиков и анимация эксперимента.

Лаборатория была разработана с использованием технологий ASP.NET на языке C#, AJAX и SVG(анимация) [2]. Технология AJAX позволяет сэкономить трафик и ускорить реакцию интерфейса за счет того, что не нужно перезагружать страницу каждый раз после ввода данных.

В дальнейшем в лабораторию могут быть добавлены другие эксперименты с соответствующим описанием, что расширит возможности ее использования.

Список литературы

1. Методы робастного, нейронечеткого и адаптивного управления / Под ред. К.А. Пупкова – М.:Изд-во МГТУ им. Н.Э. Баумана, 2002. – 744 с.
2. Парс Р., Морони Л., Гриб Д. Основы ASP.NET AJAX : Пер. с англ. / Под ред. В.А. Швеца – М.: ООО “И.Д. Вильямс”, 2009. – 288с.

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ОБРАБОТКИ ЗАКАЗОВ НА КОМПЛЕКТУЮЩИЕ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Лавров Н.А. – студент, Лукоянычев В.Г. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Обслуживанием заявок на установку вычислительной техники в университете занимается отдел эксплуатации. Существует необходимость автоматизации этого процесса для повышения эффективности работы как непосредственно отдела эксплуатации, так и всего университета в целом, ведь роль компьютерной техники в современной системе образования трудно переоценить, а своевременное решение проблем, связанных с эксплуатацией техники, будет способствовать улучшению её функционирования и устранению проблем в работе аппаратного обеспечения.

Была поставлена задача разработать программу, автоматизирующую процесс приёма и обслуживания заявок на установку вычислительной техники в университете. Тип программного обеспечения — web-приложение.

На текущий момент подобные программные продукты существуют лишь как закрытые проекты профильных компаний и слабо представлены на рынке, что говорит о востребованности программы в будущем.

В рамках решения поставленной задачи был разработан программный комплекс, состоящий из следующих модулей:

1. Пользовательский модуль
   Предоставляет пользователям автоматизированной системы возможность подачи заказов (в электронном виде) на комплектующие ВТ.
2. Модуль администратора.
   Предоставляет администратору автоматизированной системы возможность работы с базой данных заказов (добавление, редактирование, удаление данных).

РАЗРАБОТКА КОНЦЕПЦИИ ДИНАМИЧЕСКИХ ДОКУМЕНТОВ

Мажник А.А. – студент, Кантор С.А. – к.ф.-м.н, профессор
Алтайский государственный технический университет (г. Барнаул)

В последнее время всё большую актуальность приобретают задачи создания всевозможных систем автоматизации делопроизводства и документооборота, и одно из наиболее сложных и неоднозначных направлений в этом вопросе – создание и последующая обработка слабоформализованных и слабоструктурированных документов с произвольной, меняющейся в зависимости от содержимого, структурой.

Когда речь идет о документах со строго формализованной структурой, обычно используются стандартные решения, но зачастую встает задача по обработке некоего произвольного множества данных как-то связанных между собой и создание на их основе документов. Некоторые из этих данных имеют форму произвольных полнотекстовых документов, некоторые – только информацию, но с четко выраженными границами, а некоторые – наборы структурированных данных, с разбиением по фрагментам.

Сама по себе задача по обработке слабоструктурированных документов сейчас не имеет четко обоснованного и теоретического решения. Все универсальные подходы обладают теми или иными достоинствами и недостатками.

Данная работа посвящена исследованию такого типа документов и созданию программных средств для работы с ним, а цель работы – применение и внедрение созданной системы в отрасль нотариального делопроизводства и документооборота.

Для достижения указанных целей на первоначальном этапе была создана система, использующая обычные шаблоны текстовых документов, содержимое которых пользователь

мог редактировать как угодно. В современном варианте – это система редактирования логико-смысловой нагрузки документа. Т. е. программа работает с динамическими документами (live documents), подстраивая свою структуру и меня морфологические и синтаксические составляющие, по мере набора и/или изменения текста. Такие документы и система обладают следующими характеристиками и признаками:

1. Весь массив информации разделен на однозначно идентифицируемые документы.
2. Документы разделены на смысловые фрагменты – отдельные составляющие, ничем не отличающиеся от динамических документов, но находящиеся в отдельном пространстве имен и доступные для вставки с помощью специальных функций либо разметок.
3. Элементарной единицей управляемой части документа является поле.
4. Для управления поведением документа используются несложный, понятный язык разметки и подсистема полей, позволяющие легко отделить логику от содержимого и оформления.
5. Сам документ либо его любая составляющая могут быть произвольно изменены пользователем, не знакомым с языком разметки документа.
6. В режиме ввода данных документ не должен содержать элементов языка разметки, т.е. должен отображаться точно так, каким он будет на бумаге.
7. В своей структуре динамический документ формирует некую структуру машинно-обрабатываемых данных, которые в дальнейшем могут индексироваться, их можно анализировать, создавать на их основе другие документы, строить по ним статистики и т. п.
8. Для редактирования и формирования документов используется WYSIWYG редактор.

Сложность в реализации динамических документов заключается в том, что они должны работать в WYSIWYG режиме с сохранением сложного форматирования и необходимостью придерживаться стандарта RTF (Rich Text Format Specification 1.7). Для реализации таких документов используется древовидная структура, листьями которой являются атомарные поля, синхронизированные через систему управления поведением. Для того чтобы структура документа была синхронизирована с самим текстом, введен подтип поля «итератор», накапливающий в соответствии со своими правилами информацию из документа и применяя к ней различные функции (синтаксиса и морфологии). Также решена и обратная задача: по значениям итераторов перестроить структуру документа, соблюдая необходимые правила. Т.е. итератор - обычное поле, но может сам управлять деревом и наоборот листья дерева могут управлять итератором. Для обеспечения совместимости документов с другими программами структура и логика хранится в виде XML документа, в комментариях RTF ({*\fields … }, {*\fragments … } и т.д.).

Для удобства пользователей применяются специальные маркеры (‹ Unicode 0x2039: Single Left-Pointing Angle Quotation Mark и › Unicode 0x203A: Single Right-Pointing Angle Quotation Mark), чтобы подчеркнуть то, что текущий фрагмент является основным и он может влиять на остальной документ.

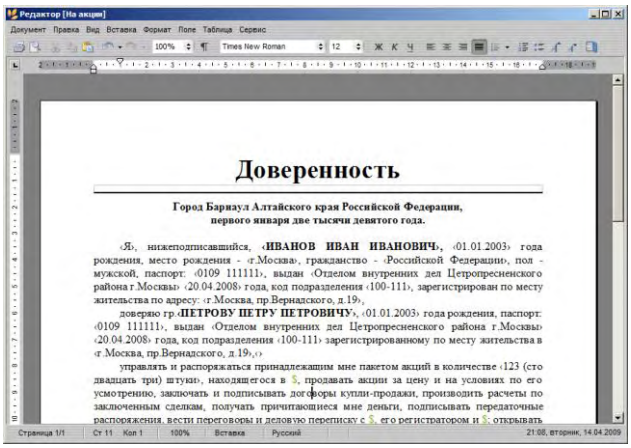
Реализация представляет собой клиент-серверное приложение, разработанное в IDE CodeGear Developer Studio (Delphi). Библиотеки, отвечающие за хранение базы данных и обрисовку RTF документов, написаны на Microsoft Visual Studio (C++). Для доступа к данным и реализации распределенной базы данных используется СУБД Firebird 2.1 либо CodeGear Interbase 7.0. Для ускорения передачи сетевого трафика применяться LZMA сжатие.

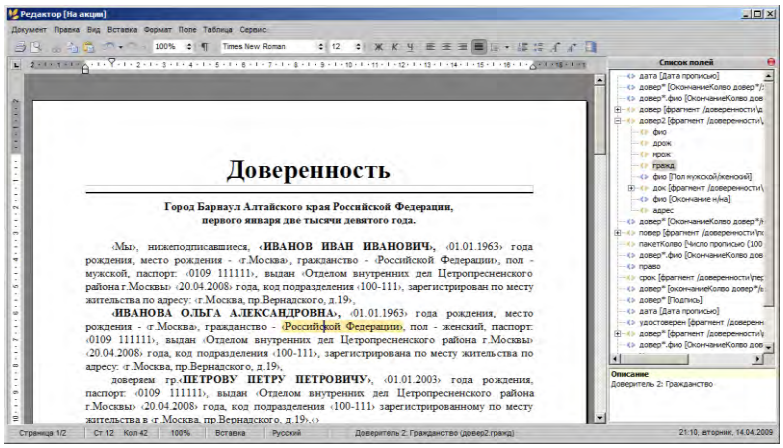
Данная концепция была применена в сфере автоматизации нотариального делопроизводства, т. к. в ней образовалась большая ниша из-за малого количества программных продуктов, полностью отвечающих требованиям работы с нотариальными документами. Она была реализована в программном продукте «Нотариат» (Свидетельство об

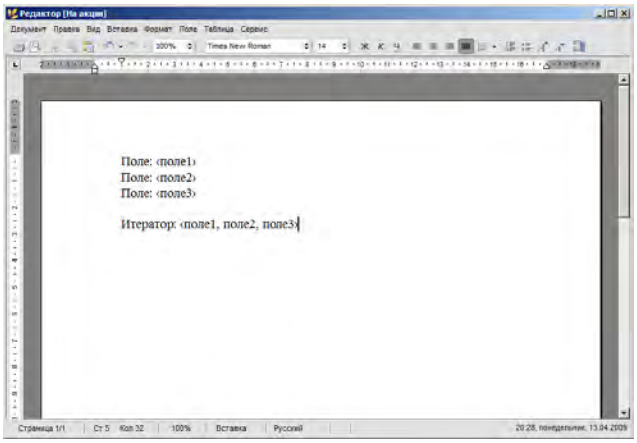
официальной регистрации программы для ЭВМ № 2006611193 от 4 апреля 2006 г.). Нотариат – это программа для автоматизации делопроизводства и документооборота нотариуса, представляющая собой сочетание полнофункционального текстового редактора, системы генерации и хранения документов.

Таким образом, предложенная концепция динамических документов может использоваться в системах автоматизации документооборота и делопроизводства, где требуется организовать работу со слабоструктурированными данными. А разработанная программа «Нотариат» уже используются в большинстве нотариальных контор Барнаула и Алтайского края для повседневной работы по обслуживанию населения. Планируется внедрение программы в местные администрации поселений Алтайского края (в связи с наделением глав в 2008 году функциями по совершению нотариальных действий), а также внедрение в нотариальных конторах других субъектов РФ.

Примеры использования







Список литературы

1. Miles L. Mathieu, Ernest A. Capozzoli «The Paperless Office: Accepting Digitized data», Troy State University, 2002
2. Kevin Craine «Excerpts from Designing a Document Strategy», Craine Communications Group
3. Le Hégaret, Philippe «The W3C Document Object Model (DOM)», World Wide Web Consortium, 2002
4. Roger T. Pédauque «Document: Form, Sign and Medium, as Reformulated for Electronic Documents», http://archivesic.ccsd.cnrs.fr, 2003
5. Bagwell Phil «Fast Functional Lists, Hash-Lists, Deques and Variable Length Arrays», EPFL, 2002

МАТЕМАТИЧЕСКИЙ АНАЛИЗ СИТУАЦИЙ В СТРАТЕГИЧЕСКИХ ИГРАХ НА ПРИМЕРЕ ИГРЫ "ТЕХАССКИЙ ХОЛДЕМ"

Маньшин В.Э. – студент, Ленюк С.В. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Игры имеют важнейшее значение в исследованиях в области Искусственного Интеллекта (ИИ) с самого начала компьютерной эры. Многие пионеры вычислительной науки провели годы, разрабатывая алгоритмы для шашек, шахмат и других стратегических игр.

Изучение настольных, карточных и других математических игр целесообразно по ряду причин. В общем, каждая из таких игр имеет несколько или все из перечисленных свойств:

- Хорошо определенные правила и логика, что позволяет относительно просто реализовать компьютерного игрока, позволяя уделить больше времени актуальным вопросам научного значения;
- Игры имеют комплексные стратегии и представляют класс сложнейших вычислительных задач;
- Игры имеют ясно определенную цель, что позволяет однозначно определить эффективность того или иного подхода и сфокусировать усилия на поиске оптимальных методов достижения этой цели;
- Игры позволяют измерять результаты путем "сталкивания" различных алгоритмов.

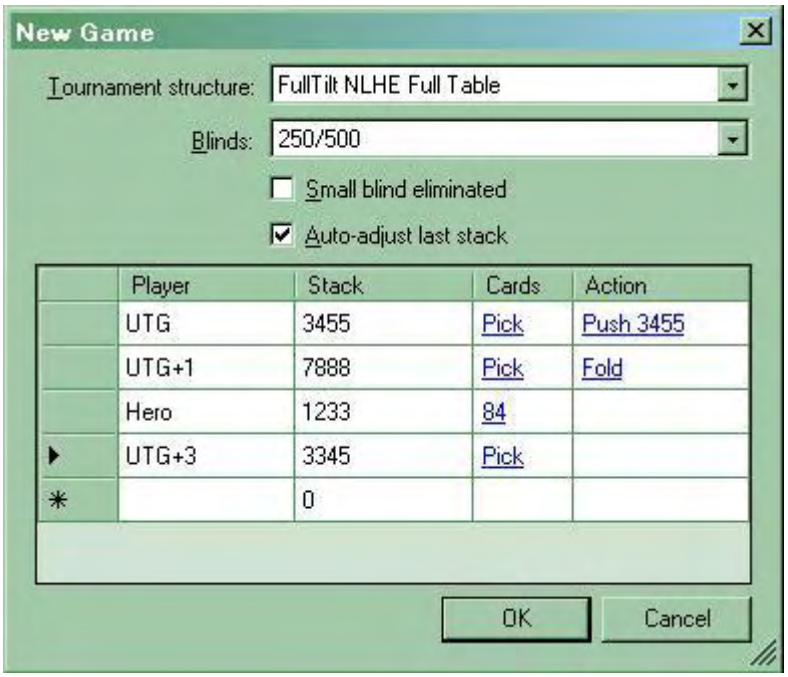
С самого основания теории игр Джоном фон Нейманом, стратегические аспекты покера не были детально изучены в компьютерной науке. Покер располагает многими характеристиками, не свойственными изучаемым ранее играм (такие как шашки и шахматы), что делает его прекрасной целью для исследований. Некоторые свойства покера в терминах теории игр:

- Покер – игра с неполной информацией. Различные формы неопределенностей встречаются постоянно. Это делает необходимым для максимизации выигрыша применение случайных смешанных стратегий;
- Покер имеет стохастические результаты. Элемент случайности (например, раздача карт) создает ситуации, которые не могут поддаваться контролю. Среди прочего, это добавляет дисперсионность в результаты игры и делает более сложным точную оценку эффективности того или иного алгоритма;
- Покер – некооперативная множественная игра. Множественность подразумевает дополнительную сложность в сравнении с парными играми.

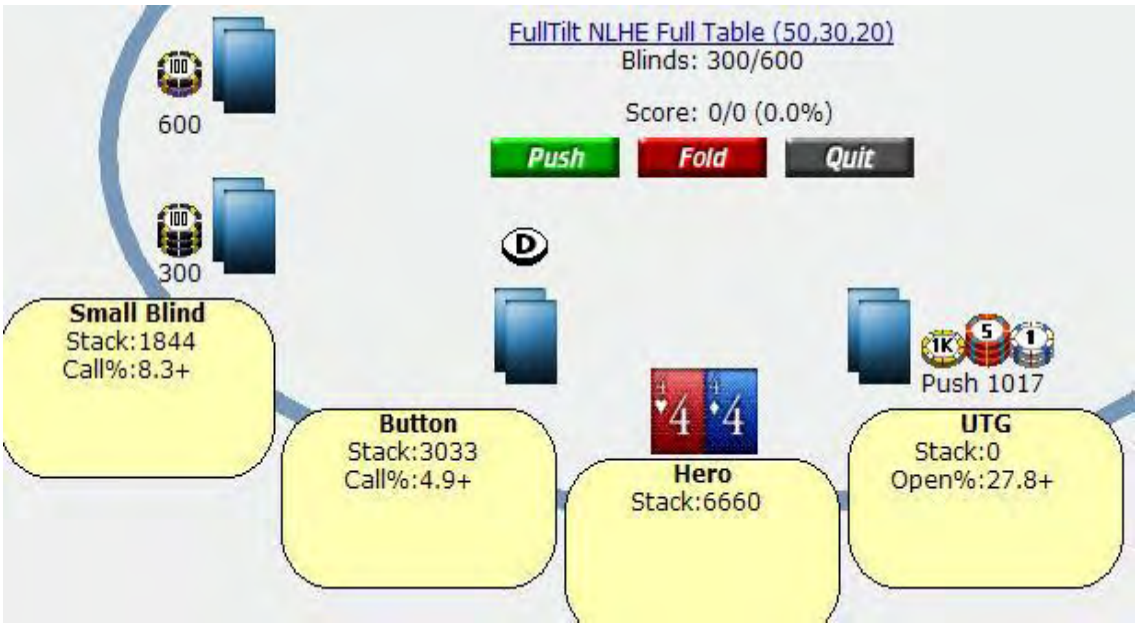
В общем, множественные стохастические игры с неполной информацией принадлежат к классу сложнейших проблем в теоретической компьютерной науке. На практике, написать программу, которая будет играть в покер в рамках правил – совсем несложно, но разработать и реализовать алгоритм, сравнимый по эффективности с компетентным покерным игроком – весьма непростая задача.

Сегодня, когда покер в России официально признан видом спорта, исследования и разработка программных продуктов в этой области становятся особенно актуальны. Разрабатываемый программный продукт будет иметь практическое приложение и обладать следующими фунциями:

- Математический анализ игровой ситуации на основании вводимых пользователем данных с возможностью наглядного и удобного, в том числе и графического, представления результатов анализа пользователю;



- Настраиваемый режим обучения. В данном режиме пользователю предлагается принимать решения в смоделированных программой ситуациях. Ведется подсчет верных и неверных ответов с возможностью последующего просмотра и анализа ситуаций, в которых были допущены ошибки. Гибкая система настроек моделирования позволяет пользователю желаемый спектр игровых ситуаций для тренировки.



В силу специфический особенностей, покер отлично подходит для игры в сети Интернет. Каждый день сотни тысяч спортсменов со всего мира проводят миллионы игр, а также участвуют в крупных соревнованиях. В связи с этим, разрабатываемый программный продукт будет включать ряд следующих возможностей:

- Возможность автоматической загрузки игровых ситуаций для последующего анализа из баз данных крупнейших Интернет – порталов, организующих соревнования по покеру;
- Возможность детальной настройки параметров анализа и моделирования для полного соответствия всем правилам соревнований на разлиных игровых площадках;
- Возможность экспорта результатов анализа и обучения в распространенные форматы для последующего использования.

Таким образом, в разрабатываемой программе предусмотрен полный набор функций и настроек для того, чтобы она могла стать отличным тренажером для спортсмена, а интуитивно понятный интерфейс и поддержка многоязычности сделают ее максимально легкой в освоении.

Список литературы

1. Оуэн Г. Теория игр. / Г. Оуэн – 2-е изд. – М.: Едиториал УРСС, 2004. – 216 с.
2. N. Bard. Using State Estimation for Dynamic Agent Modelling. [Электронный ресурс] – Режим доступа: http://www.cs.ualberta.ca/~games/poker/papers/bard.msc.html.
3. M. Johanson, M. Zinkevich, M. Bowling. Computing Robust Counter-Startegies. [Электронный ресурс] – Режим доступа: http://www.cs.ualberta.ca/~games/poker/papers/NIPS07-rnash.html.
4. M. Johanson, M. Bowling. Data Biased Response Counter Strategies. [Электронный ресурс] – Режим доступа: http://www.cs.ualberta.ca/~games/poker/papers/AISTATS09.html.

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ВЫДЕЛЕНИЯ ОБЛАСТЕЙ С ОДНОРОДНОЙ ТЕКСТУРОЙ ПРИ ОБРАБОТКЕ КОСМИЧЕСКИХ СНИМКОВ

Масков И.Г. – студент, Андреева А.Ю. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

При анализе изображений важной их характеристикой служит текстура, которая присутствует во всех изображениях, начиная с изображений, получаемых с помощью авиации и спутниковых устройств и кончая микроскопическими изображениями в биомедицинских исследованиях. Сегментация текстур важна в автоматизации обработки космических и аэрофотоснимков земли. Например, космические и самолетные снимки используются в лесоводстве для анализа структуры и динамики лесного покрова в целях мониторинга и инвентаризации. С помощью космических снимков определяются обширные регионы с одинаковыми почвенно-климатическими условиями, выделяются области, занятые лесными угодьями. Подобласти регионов обследуются аэрофотосъемкой. На этом масштабном уровне главным признаком различения участков выступает текстура изображения, закономерно отражающая внутреннее строение лесных сообществ.Текстурных свойств насаждений обычно бывает достаточно, чтобы различить тип леса и его возраст. В связи с частым возникновением подобных задч возникает необходимость в системах сегментации областей с постоянной текстурой на растровых изображениях.

Была поставлена задача разработать программный продукт, который должен стать основой для построения узконаправленных систем анализа снимков. В качесвте базовых методов было решено использовать метод блока фильтров [1] и статистические признаки Харалика [2].

На текущий момент подобные программные продукты существуют лишь как закрытые проекты профильных компаний (занимающихся картографией, геодезией и т. д.) и слабо представлены на рынке. Последнее говорит востребованности программы в будущем.

В рамках решения поставленной задачи был разработан программный комплекс в котором реализованы:

- метод блока фильтров;
- статистический метод, основанный на использовании матрицы смежности значений яркости;
- возможность работы с различными растровыми форматами изображений, в частности с BMP, PNG, JPG;
- возможность настройки всех основных параметров методов выявления признаков текстуры;
- демонстрация промежуточных результатов работы методов.

По результатам сегментации тестовых снимков выполнен анализ качества сегментации областей с однородной текстурой каждым алгоритмом.

Список литературы

1. Форсайт, Д. Компьютерное зрение. Современный подход / Д. Форсайт, Д. Понс : A Modern Approach. — М.: «Вильямс», 2004. — С. 928.
2. Харалик Р.М. Статистический и структурный подход к описанию текстур // ТИИЭР, 1979 г., т. 67, № 5, с. 98—120
3. Джордж, С. Компьютерное зрение / С. Джордж , Л. Шапиро — М.: Бином. Лаборатория знаний, 2006. — С. 752.

РАЗРАБОТКА ВЫСОКОНАГРУЖЕННЫХ ПРИЛОЖЕНИЙ НА ПРИМЕРЕ ФРЕЙМВОРКА ДЛЯ ПРОВЕДЕНИЯ СОРЕВНОВАНИЙ ПО ПРОГРАММИРОВАНИЮ

Могозов А.А. – студент, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Высоконагруженные масштабируемые приложения широко востребованы в современном мире с высокоскоростными каналами связей и большими объемами обрабатываемых данных. Однако, разработка подобных систем – не простая задача. Чтобы приложение было способно устойчиво обрабатывать многочисленные запросы в единицу времени, необходимо тщательно проектировать каждый компонент системы, а также взаимосвязь между ними.

Чтобы понять принципы проектирования подобных систем, необходимо выяснить, что же такое распределенная система. Часто при определении распределенной системы во главу угла ставят разделение ее функций между несколькими компьютерами. При таком подходе распределенной является любая вычислительная система, где обработка данных разделена между двумя и более компьютерами. Основываясь на определении Э. Таненбаума, несколько более узко распределенную систему можно определить как набор соединенных каналами связи независимых компьютеров, которые с точки зрения пользователя некоторого программного обеспечения выглядят единым целым. Такой подход к определению распределенной системы имеет свои недостатки. Например, все используемое в такой распределенной системе программное обеспечение могло бы работать и на одном единственном компьютере, однако с точки зрения приведенного выше определения такая система уже перестанет быть распределенной. Поэтому понятие распределенной системы, вероятно, должно основываться на анализе образующего такую систему программного обеспечения.

В рамках поставленной проблемы важно взаимодействие составных частей

программного обеспечения, образующего распределенную систему. Рассмотрим некое типичное приложение, которое в соответствии с современными представлениями может быть разделено на следующие логические уровни: пользовательский интерфейс (ИП), логика приложения (ЛП) и доступ к данным (ДД), работающий с базой данных (БД). Пользователь системы взаимодействует с ней через интерфейс пользователя, база данных хранит данные, описывающие предметную область приложения, а уровень логики приложения реализует все алгоритмы, относящиеся к предметной области.

Поскольку на практике разных пользователей системы обычно интересует доступ к одним и тем же данным, наиболее простым разнесением функций такой системы между несколькими компьютерами будет разделение логических уровней приложения между одной серверной частью приложения, отвечающим за доступ к данным, и находящимися на нескольких компьютерах клиентскими частями, реализующими интерфейс пользователя. Логика приложения может быть отнесена к серверу, клиентам, или разделена между ними.

Таким образом, появляется проблема осуществления удалённого взаимодействия. Для её решения существует несколько технологий, таких как Веб-службы, Java RMI, .NET Remoting.

Технология RMI разрабатывалась специально для Java. Проведенные автором исследования показывают, что RMI позволяет реализовать прозрачное для разработчика удалённое взаимодействие без использования дополнительных средств. В то же время, основным протоколом взаимодействия является TCP/IP, передача данных ведется через не стандартные порты, что создает дополнительные трудности при использовании брандмауэров.

Remoting разрабатывалась для платформы .NET, ее архитектура схожа с RMI. Исследования технологии выявили неустойчивое функционирование в гетерогенной среде. Возможно, в будущих версиях разработчикам удастся это исправить, и Remoting займет достойное место в области распределенного взаимодействия.

Во время проведенного автором исследования веб-служб были выявлены следующие достоинства:

– Веб-службы обеспечивают взаимодействие программных систем независимо от платформы;

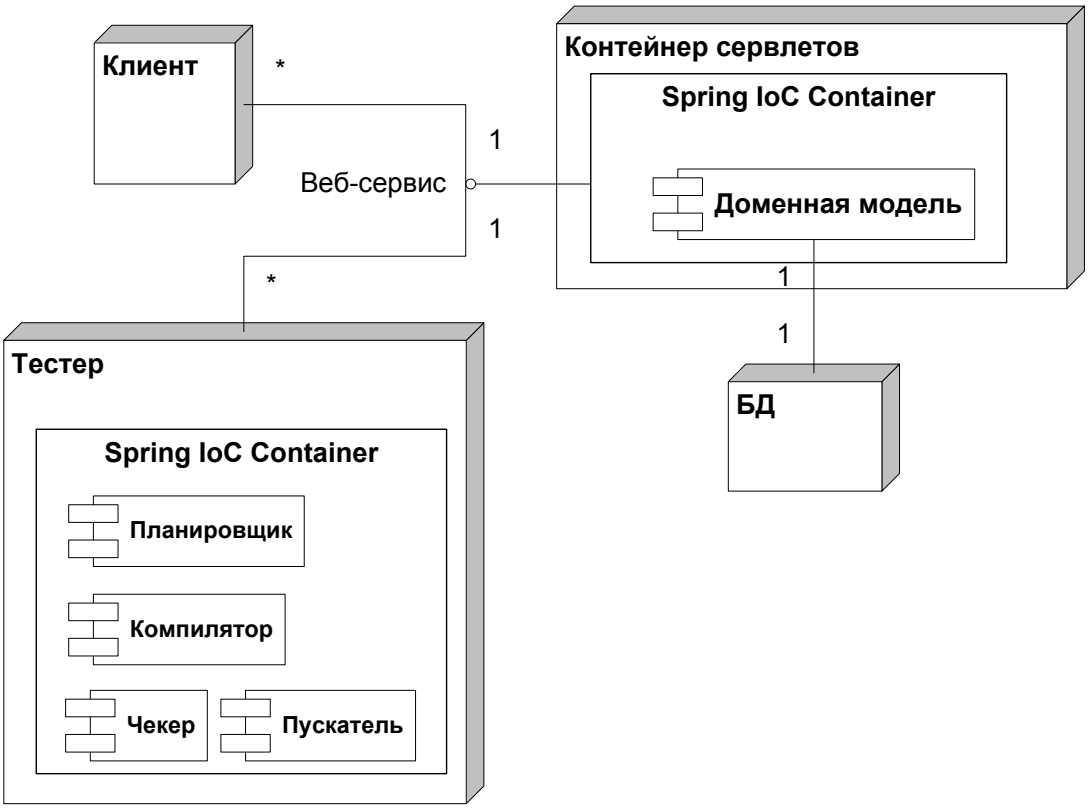
– Веб-службы основаны на базе открытых стандартов и протоколов, благодаря использованию XML достигается простота разработки и отладки веб-служб;

– Использование интернет-протокола HTTP обеспечивает взаимодействие программных систем через межсетевой экран.

Основным недостатком веб-служб является меньшая производительность и больший размер сетевого трафика по сравнению с технологиями RMI, CORBA, DCOM за счет использования текстовых XML-сообщений. Тем не менее, при наличии современных каналов связи, объем трафика, генерируемого веб-службами, можно считать ничтожно малым.

Из технологий обеспечения удалённого взаимодействия автором были выбраны веб-службы, поскольку в данном случае нет привязки к конкретному языку программирования. В то же время при использовании веб-служб совместно с Java становятся доступны такие решения, как Spring Framework.

Архитектура предлагаемой системы выглядит следующим образом:



Spring обеспечивает легковесное решение для создания приложений уровня предприятия, плюс ко всему поддерживается:

– возможность использования декларативного управления транзакциями;

– удаленный доступ к бизнес процессам через RMI, веб-сервисы;

– множество функций для работы с планировщиком и почтой;

– широкая поддержка БД и ORM.

Использование Spring в разрабатываемой системе влечет за собой уменьшение связности компонентов между собой, поскольку ядро Spring реализует шаблон проектирования Dependency Injection или Inversion of Control. Данный принцип превращает систему в каркас – чтобы добавить новый компонент, необходимо реализовать предлагаемые интерфейсы, изменить конфигурационный файл контекста приложения, а для корректного взаимодействия с базой данных – придерживаться существующей доменной модели.

Чтобы избавить разработчиков от утомительного программирования взаимодействия с базой данных, в каркасе использован Hibernate. Это библиотека для языка программирования Java, предназначенная для решения задач объектно-реляционного проецирования (object-relational mapping — ORM). Она представляет собой свободное программное обеспечение с открытым исходным кодом (open source), распространяемое на условиях GNU Lesser General Public License. Данная библиотка предоставляет лёгкий в использовании каркас для отображения объектно-ориентированной модели данных в традиционные реляционные базы данных.

В современной действительности многие университеты проводят соревнования по различным дисциплинам, в том числе и по программированию. На стандартном соревновании участникам предлагается на выбор несколько задач. Решениями являются исходные коды программ, решающие данные задачи. Чтобы организаторы могли гарантировать достоверность результатов соревнований, каждое решение должно быть

проверено на наборе тестов. Проверка заключается в запуске решения на каждом тесте и сверке результата, полученного участником, с эталоном. Если бы проверка происходила вручную, то данный процесс занимал бы очень длительное время, не исключены были бы ошибки проверки. Помимо этого, значительных усилий требует подведение итогов соревнования. Поэтому, чтобы автоматизировать обработку решений и генерацию результатов, требуется компьютерная программа. В этом случае время проверки сводится к минимуму, исключаются все ошибки в результатах. Также для удобства работы клиенты должны иметь возможность посылать решения удалённо.

Таким образом, в разрабатываемой системе соблюдены все требования к распределенным приложениям, и данный каркас подходит для проведения соревнований любого уровня – от внутривузовских до международных студенческих контестов, параллельно проходящих в нескольких городах. Например, при увеличении количества необработанных решений участников, существует возможность запуска еще одного тестирующего модуля, а чтобы реализовать специфичные для соревнования системы оценки решения и подсчета итоговых результатов, необходимо добавить собственные модули.

Список литературы

1. Таненбаум Э. Распределенные системы. Принципы и парадигмы / Э. Таненбаум, М. ван Стеен. – СПб.: Питер, 2003. – 877 с.
2. Spring framework [Электронный ресурс] – Режим доступа: http://www.springframework.org/
3. Hibernate [Электронный ресурс] – Режим доступа: http://www.hibernate.org
4. Fowler M. Inversion of Control Containers and the Dependency Injection pattern [Электронный ресурс] – Режим доступа: http://martinfowler.com/articles/injection.html

АВТОМАТИЗАЦИЯ ФОРМИРОВАНИЯ ДОКУМЕНТАЦИИ ПО ОРГАНИЗАЦИИ УЧЕБНОГО ПРОЦЕССА НА ВЫПУСКАЮЩЕЙ КАФЕДРЕ

Назарова З.Ю. – студентка, Сучкова Л.И. – к.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В настоящее время на выпускающей кафедре университета при организации учебного процесса обрабатывается большое количество входной информации, и формируются различные виды документов планового, учётного и контрольного характера. Данный процесс является очень трудоёмким и монотонным, поэтому необходима автоматизация формирования учебной документации кафедры.

Разработан программный продукт, позволяющий упростить и ускорить процесс распределения и расчета учебной нагрузки, осуществлять контроль правильности составления графиков СРС, а так же обеспечить хранение архива документов. Для хранения и обработки информации используется клиент-серверная СУБД MySQL, которая физически располагается на сервере кафедры. Программный продукт позволяет изменять настройки для расчёта нагрузки преподавателей, заносить данные для расчёта в базу данных, просматривать результаты расчётов и формировать на их основе различные виды документов: расчёт учебной нагрузки ППС кафедры на учебный год; ведомость учета выполненной учебной, научной, методической и воспитательной работы; учебные поручения преподавателям кафедры; семестровые графики учебного процесса, графики СРС.

В рамках решения поставленной задачи реализованы следующие возможности:

1. Спроектирована структура базы данных для хранения информации, относящейся к расчёту и распределению нагрузки, контролю выполнения нагрузки и заполнению графиков учебного процесса;

2. Разработан программный модуль, позволяющий выполнять следующие виды работ:

- ввод в базу данных, редактирование и удаление из базы данных информации о преподавателях, факультетах, кафедрах, специальностях, группах, дисциплинах, видах занятий;
- распределение учебной нагрузки между преподавателями кафедры, в том числе деление часов дисциплины между несколькими преподавателями. Расчет учебной нагрузки выполняется как с учетом учебного плана специальности (специальностей), выпускаемых кафедрой, так и с учетом учебных поручений с других кафедр и деканатов;
- расчёт общей нагрузки на преподавательский состав кафедры и нагрузки на каждого преподавателя в отдельности в часах и в ставках. Достоинством программы является легкость распределения нагрузки по одной дисциплине, читаемой потоку студентов, между различными преподавателями. Возможно распределение по подгруппам и по видам занятий (лекции, практики, лабораторные работы, курсовое проектирование);
- корректировка настроек программы: нормативов времени на разные виды занятий, размеров в часах ставки штатного преподавателя и максимально допустимой нормы времени в рамках почасовой оплаты;
- формирование выходной документации для обеспечения учебного процесса на кафедре: ведомость расчёта учебной нагрузки на год; ведомость учета выполненной нагрузки (форма 6) по семестрам и за год; учебные поручения преподавателям кафедры; семестровые графики учебного процесса, графики СРС для студентов;
- архивирование устаревшей информации для последующего хранения в течение указанного времени и извлечения из архива нужных сведений по требованию.

О 2-ГРУППАХ ПОРЯДКА 64 С ГРУППОЙ АВТОМОРФИЗМОВ ТОГО ЖЕ ПОРЯДКА

Ольхин М.А. – студент, Ильин В.И. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Из GAP – источника [1] известно, что существует 267 неизоморфных групп порядка 64, то есть, состоящих из 64 элементов. Из них только 3 группы: $G_1 = 64 / 31$, $G_2 = 64 / 50$, $G_3 = 64 / 51$ имеют порядки, совпадающие с порядками их групп автоморфизмов, то есть $|G_i| = |Aut(G_i)| = 64$, $i = 1, 2, 3$. Здесь, в числителе указан фиксированный порядок группы N=64, а в знаменателе указаны номера групп по GAP-овской нумерации p-групп фиксированного порядка N. Целью работы является сравнение строения группы и строения ее группы автоморфизмов. Работа связанна с вопросом 15.29 из Коуровской тетради [2] : (A. Mann). Диэдральная группа порядка 8 изоморфна своей группе автоморфизмов. Есть ли другие нетривиальные конечные p-группы с таким же свойством?

Из GAP-системы извлечены следующие копредставления и свойства групп G_i :

```
G1=64/31
(a1,a2,a3,a4,a5,a6;a1^2=a4,a2^2=1,a3^2=a6,a4^2=a5,a5^2=a6,a6^2=1,
[a1,a2]=a3*a6,[a1,a3]=1,[a1,a4]=1,[a1,a5]=1,[a1,a6]=1,
[a2,a3]=a6,[a2,a4]=a6,[a2,a5]=1,[a2,a6]=1,
[a3,a4]=1,[a3,a5]=1,[a3,a6]=1,
[a4,a5]=1,[a4,a6]=1,
[a5,a6]=1)

Таблица порядков элементов:
1 elemens of order 1
7 elements of order 2
8 elements of order 4
16 elements of order 8
32 elements of order 16
```

```
28 conjugacy classes
Centre generated by [ a3*a4, a5, a6 ] , type 8/1
Commutator subgroup generated by [ a3, a6 ] , type 4/1
Automorphism Group of size 64
Inner automorphism group type 8/4
Nilpotency class 3
Lower Central Series:  64/31 --- 4/1 --- 2/1 --- 1/1
```

G_2=64/50

```
(a1,a2,a3,a4,a5,a6;a1^2=a3,a2^2=1,a3^2=a4,a4^2=a5,a5^2=a6,a6^2=1,
[a1,a2]=1,[a1,a3]=1,[a1,a4]=1,[a1,a5]=1,[a1,a6]=1,
[a2,a3]=1,[a2,a4]=1,[a2,a5]=1,[a2,a6]=1,
[a3,a4]=1,[a3,a5]=1,[a3,a6]=1,
[a4,a5]=1,[a4,a6]=1,
[a5,a6]=1)
```

Таблица порядков элементов:

```
1 element of order 1
3 elements of order 2
4 elements of order 4
8 elements of order 8
16 elements of order 16
32 elements of order 32

ABELIAN

group is isomorphic to 2/1 x 32/1
```

G_3=64/51

```
(a1,a2,a3,a4,a5,a6;a1^2=a3,a2^2=1,a3^2=a4,a4^2=a5,a5^2=a6,a6^2=1,
[a1,a2]=a6,[a1,a3]=1,[a1,a4]=1,[a1,a5]=1,[a1,a6]=1,
[a2,a3]=1,[a2,a4]=1,[a2,a5]=1,[a2,a6]=1,
[a3,a4]=1,[a3,a5]=1,[a3,a6]=1,
[a4,a5]=1,[a4,a6]=1,
[a5,a6]=1)
```

Таблица порядков элементов:

```
1 element of order 1
3 elements of order 2
4 elements of order 4
8 elements of order 8
16 elements of order 16
32 elements of order 32

40 conjugacy classes
Centre generated by [ a3, a4, a5, a6 ] , type 16/1
Commutator subgroup generated by [ a6 ] , type 2/1
Automorphism Group of size 64
Inner automorphism group type 4/2
Nilpotency class 2
Lower Central Series:  64/51 --- 2/1 --- 1/1
```

Как видно, в копредставлениях используются степенные и коммутаторные соотношения и выглядят они довольно громоздко. Отметим сразу, что абелев, наиболее простой случай, нами пока не исследован. Используя преобразования Титце, получены более простые представления следующего вида:

$$G_1=64\ /\ 31=< a_1,\ a_2,\ a_3 \mid a_1^{16} = a_2^{2} = a_3^{4} = [a_1,a_3]=1,\ [a_1,a_2]=a_3^{3},\ [a_2,a_3]=a_3^{2}=a_1^{8},\ [a_2,a_1^{2}]=a_1^{8},\ [a_2,a_1^{4}]=[a_2,a_1^{8}]=[a_3,a_1^{2}]=[a_3,a_1^{4}]=1 >;$$

$$G_3=64\ /\ 51=< a_1,\ a_2 \mid a_1^{32}=a_2^{2}=1,\ [a_1,a_2]=a_1^{16},\ [a_2,a_1^{2}]=[a_2,a_1^{4}]=[a_2,a_1^{8}]=[a_2,a_1^{16}]=1 >.$$

Любой элемент g из G_1 или G_3 можно представить в виде:

$g=a_1^{\alpha}.a_2^{\beta}.a_3^{\gamma}$, где $0 \le \alpha \le 15$, $0 \le \beta \le 1$, $0 \le \gamma \le 3$ или $g=a_1^{\alpha}.a_2^{\beta}$, где $0 \le \alpha \le 31$, $0 \le \beta \le 1$ соответственно. Занумеровав элементы групп G_1 и G_3 , используя соотношения, были построены регулярные представления этих групп, из которых затем были получены таблицы умножения элементов, представленных натуральными числами. Для построения автоморфизмов использовались следующие соображения:

Любой автоморфизм должен:

1. Сохранять порядки элементов,
2. Переводить систему образующих в систему образующих;
3. Новая система образующих должна удовлетворять всем определяющим соотношениям группы.

Все вышеизложенные идеи были реализованы с помощью компьютерных программ. В результате вычислений получены следующие таблицы порядков элементов групп автоморфизмов $Aut(G_1)$ и $Aut(G_3)$:

Таблица порядков элементов $Aut(G_1=64 / 31)$:
1 элемент порядка 1
23 элемента порядка 2
40 элементов порядка 4;

Таблица порядков элементов $Aut(G_3=64 / 51)$:
1 элемент порядка 1
15 элементов порядка 2
16 элементов порядка 4
32 элемента порядка 8;

Из сравнения таблиц порядков элементов групп G_1 и G_3 и их групп автоморфизмов видно, что они не изоморфны друг другу. Отождествление групп $Aut(G_1)$ и $Aut(G_3)$ с одной из 267 групп порядка 64 пока не получено.

Список литературы

1. GAP – Groups, Algorithms, Programming- a System for Computational Discrete Algebra, http: //www.gap-system.org/.
2. Нерешенные вопросы теории групп. Коуровская тетрадь, 16-е изд., Новосибирск, Ин-т матем. СО РАН, 2006, http: // www.math.nsc.ru / ~alglog.

АВТОМАТИЗИРУЮЩАЯ СИСТЕМА МОДЕЛИРОВАНИЯ И АНАЛИЗА ДИНАМИЧЕСКИХ СИСТЕМ НА ОСНОВЕ СЕТЕЙ ПЕТРИ

Отморский С.А. – студент, Крючкова Е.Н. – к.ф-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В современном усложняющемся мире все большую роль начинает играть возможность моделирования и симулирования работы больших динамических дискретных систем. В связи с этим потребовалась такая формальная модель на которой с одной стороны были бы видны многие ошибки связанные с проектированием системы, а с другой стороны модель должна быть достаточно проста (чтобы построение модели было намного проще чем реализация системы). Одна из самых популярных таких моделей – сети Петри.

Сетью Петри называется набор:

$$N = (P, T, F, W, M_0)$$

где P – не пустое множество элементов сети, называемых местами (places)

T – не пустое множество элементов сети, называемых переходами (transitions)

$F \subseteq P \times T \cup T \times P$ – множество дуг из мест в переходы и из переходов в места.

W – кратность дуг, функция сопоставляющая каждой дуге число больше 0.

M_0 – начальная разметка, функция сопоставляющая каждому месту некоторое число – разметку места[1].

Работу сети можно представить как последовательность срабатывания переходов. Переход может сработать если число фишек в местах из которых есть дуги ведущие в переход больше или равно кратности соответствующих дуг. Если есть несколько переходов, которые могут сработать, то срабатывает выбранный случайно переход. Срабатывание перехода уменьшает количество фишек в местах, из которых есть дуги ведущие в переход, и увеличивает число фишек в местах, в которые ведут дуги из перехода. Кратность дуги определяет на сколько необходимо уменьшить или увеличить количество фишек в месте[1].

К сожалению на сегодняшний день не существует программного обеспечения, которое обладало всеми из следующих возможностей одновременно:

1) Удобным графическим редактором;
2) Способностью моделировать поведение сети;
3) Функцией анализа свойств сети.

Существуют мощные графические редакторы сетей Петри, но в них не предусмотрены функции моделирования и анализа. Главным недостатком ПО способного моделировать работу сети является их редактор, реализующий лишь необходимый для редактирования минимум функций. Ни одно из существующего на сегодняшний день ПО не способно анализировать сети Петри.

В связи с этим было принято решение разработать систему позволяющую моделировать и анализировать сети Петри, а также обладающую удобным графическим редактором сетей.

Программа состоит из нескольких модулей, каждый из которых решает свою подзадачу. Архитектура системы представлена на рисунке 1:

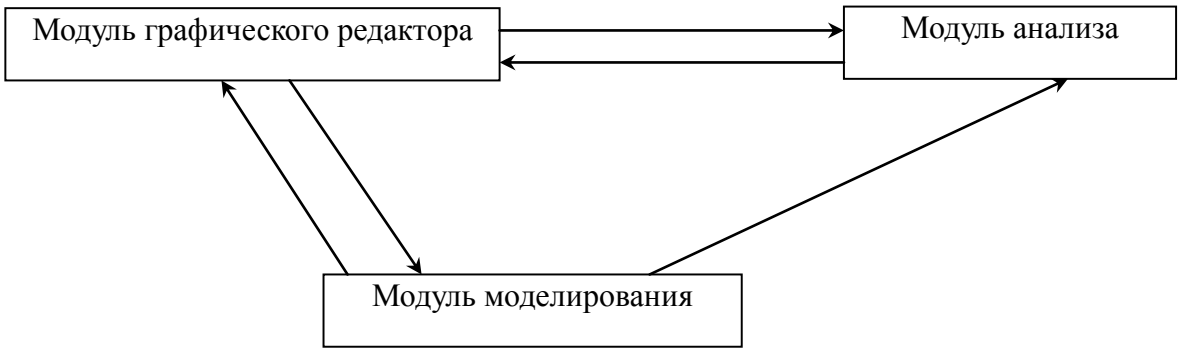


Рисунок 1 Архитектура системы

Графический редактор предназначен для взаимодействия с пользователем. Он представляет пользователю данные в стандартной для сетей Петри форме. А именно: места обозначены кружками с фишками внутри (числом, если фишек слишком много), количество фишек означает разметку места, переходы прямоугольниками, т.е. в форме принятой в теории сетей Петри. Пользователь может редактировать сеть. Интерфейс графического редактора достаточно стандартен, у пользователя не возникнет проблем с обучением если он уже работал с любым UML редактором (например MS Visio, AgroUML).

Модуль моделирования позволяет наглядно увидеть различные варианты поведения сети при запуске из начальной разметки. С некоторым временным интервалом он выбирает один из переходов, среди тех которые могут сработать из текущей. После чего происходит замена текущей разметки, на разметку, которая получится после срабатывания перехода в соответствии с правилом срабатывания переходов. Процедура выбора, в зависимости от

настроек, может выбирать переход случайным образом или предоставить выбор пользователю.

Алгоритмы модуля анализа позволяют определить, обладает ли сеть следующими свойствами[2]:

1. Сохраняемость. Сеть Петри называется сохраняемой, если сумма по всем местам всех разметок, достижимых из начальной (включая начальную) постоянна;
2. Ограниченность. Сеть Петри называется ограниченной, если существует такое число К, что для любой разметки достижимой из начальной разметка любого места не превышает К;
3. Безопасность. Частный случай ограниченности с числом К = 1;
4. Достижимость разметки. Сеть Петри может достигнуть заданную разметку, если существует такая последовательность срабатываний переходов, которая переводит сеть из начальной разметки в заданную;
5. Живость. Сеть называется живой, если любой переход может сработать в процессе ее функционирования.

Графический редактор и алгоритмы работают в параллельных потоках, что позволяет пользователю продолжить работу с редактором, во время работы алгоритмов.

Программа находится в стадии разработки.

Список литературы

1. Котов В.Е. Сети Петри. - М.: Наука. Главная редакция физико-математической литературы. - 1984. - 160 с.
2. Сети Петри – Википедия. - [Электронный ресурс] – Режим доступа: / http://ru.wikipedia.org / wiki/Сети_Петри

АВТОМАТИЗАЦИЯ РАССЧЕТА СТАЛЬНОЙ КОЛОННЫ

Парамзин М.Н. – студент
Бубнова Н.Д. – ст.преподаватель, Бусыгина Г.М. – к.э.н, доцент кафедры СК
Алтайский государственный технический университет (г. Барнаул)

С развитием технологий возведения зданий с металлическим каркасом появилась необходимость в проектах, позволяющих с наибольшей эффективностью применять прочностные характеристики металла. Одной из задач является нахождение наиболее экономичного варианта формирования составного сечения металлической колонны. Решение этой задачи является сложным процессом, требующим больших вычислений и многократных итераций, поэтому возникает необходимость в автоматизации этого процесса.

Разрабатываемый программный комплекс «Расчет стальной колонны» состоит из модулей, предназначенных для решения следующих задач:

- выбор типов металлопроката, используемых в сечении;
- выбор из нормативных справочников характеристик проката;
- вычисление геометрических характеристик сечения;
- выбор из справочника характеристик стали;
- вычисление геометрических параметров колонны;
- выбор коэффициентов из таблиц СНиПа методами интерполяции;
- определение нормальных напряжений в колонне;
- формирование вывода о соответствии подобранного сечения заданной нагрузке.

Исходными данными для расчета является нагрузка, передаваемая на колонну - продольная сила и изгибающий момент, длина колонны, геометрические размеры планки и номер стали. Составное сечение колонны задается пользователем из имеющихся типов

проката, примером которых являются горячекатаные швеллера, двутавры и т.д. Характеристики сортаментов и параметры стали хранятся в базе данных, являющейся составной частью разрабатываемого ПИО. По выбранному прокату из базы данных извлекается информация, необходимая для расчета геометрических характеристик сечения, таких как площадь, координаты центра тяжести, радиусы инерции, моменты инерции, а также радиусы инерции ветвей.

После последующего вычисления геометрических параметров возникает необходимость расчета коэффициента продольного изгиба центрально-сжатых элементов, а также коэффициента для проверки устойчивости внецентренно сжатых стержней. Значения коэффициентов определяются на основе линейной интерполяции таблиц 72 и 74 СНиП II-23-81* Стальные конструкции. Правильность выбора метода линейной табличной интерполяции подтверждается инженерной практикой.

Вычисленные напряжения в колонне проверяются на соответствие расчетному сопротивлению стали, что дает возможность сделать вывод о том выдержит ли колонна данную нагрузку. Используя ПИО, проектировщик может выполнить расчет для различных видов сечений и выбрать из них тот, который наилучшим образом соответствует техническому заданию.

Программный комплекс реализуется в среде Builder C++. Таблицы базы данных создаются средствами FoxPro. Работа с базой данных выполняется из программы.

Список литературы

1. ГОСТ 8240-97 Швеллеры стальные горячекатаные. Сортамент. Постановление Госстандарта России от 05.04.2001 N 166-ст ГОСТ от 05.04.2001 N 8240-97

АВТОМАТИЗАЦИЯ КЛАССИФИКАЦИИ ТИПА МОРФОЛОГИИ КЛЕТКИ МЕТОДОМ КЛАСТЕРНОГО АНАЛИЗА

Перепелица Н.В. – студентка, Андреева А.Ю. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

В настоящее время возможности интеллектуального анализа изображений с помощью компьютеров оставляют желать большего. А способы углубленной их обработки и распознавания требуются во многих областях, включая медицину. Поэтому особенно актуальны экспертные системы, опирающиеся на базы данных, включающие изображения, для поиска и распознавания заданных объектов, а так же их анализа.

Программное обеспечение по обработке медицинских изображений и вычисления для определенного типа клеток гистологических характеристик строго ограничено типом обрабатываемых клеток. Особенно мало исследованным остается вопрос взаимосвязи между морфологией клетки и показателями дисфункций эндотелия.

Для разрешения данной проблемы был разработан программный продукт, позволяющий хранить и вести базу карточек пациентов, включающих галерею снимков клеток эндотелия больного, автоматизировать аппарат обработки и классификации изображений клеток эндотелия пациента, упростить и ускорить работу при проведении исследований.

Для хранения и обработки информации используется СУБД MySQL, база данных физически располагается на том же компьютере, что и работающее приложение. С использованием разработанного программного продукта врач-исследователь может вести базу данных пациентов, используя галерею снимков клеток эндотелия и реализованный функционал по обработки изображений эндотелиоцитов, проводить анализ клеток и классификацию, выставлять и изменять в ходе проведения исследований диагноз пациенту.

В рамках решения поставленной задачи реализованы следующие возможности:

1. Спроектирована структура базы данных для хранения необходимой в исследовательской работе информации;

2. Разработан программный комплекс, позволяющий проводить автоматическую классификацию изображений клеток эндотелия человека.

Программный комплекс по работе с изображениями клеток эндотелия пациентов реализует следующие возможности:

- создание карточки пациента, внесение личных данных больного и набора фотоснимков его клеток эндотелия для дальнейшей обработки;
- нормирование изображения клетки;
- выделение контура клетки;
- задание исследуемых форм клеток и характерных для каждой из форм вариантов контуров;
- построение нейросети для классификации клетки по форме;
- кластеризация формы клетки с целью установки взаимосвязи между морфологией клетки и показателями дисфункции эндотелия;
- выставление диагноза, выводов проведенной обработки.

Так как на данном этапе еще не установлены корреляционные зависимости между морфологией клетки и показателями дисфункции эндотелия, полученные в результате кластерного анализа данные важны для проведения исследовательских работ в области выявления дисфункции эндотелия и лечения сердечно-сосудистых заболеваний.

ЧИСЛЕННОЕ МОДЕЛИРОВАНИЕ ОРБИТАЛЬНЫХ ВОЗБУЖДЕНИЙ ЭЛЕКТРОНОВ В КЛАСТЕРАХ

Попов В.В. – аспирант, Кантор С.А. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Современная теория несамосопряженных операторов еще далека от той степени завершенности, которая позволяет включать ее в физические учебники. Уже поэтому создание такой теории является весьма актуальной задачей. Не менее важным является и применение этой теории к описанию открытых систем, систем с поглощением и излучением. Применение несамосопряженных операторов позволяет на качественно новом уровне выполнять моделирование процессов, протекающих в открытых системах, в рамках единой *ab initio* схемы рассчитывать большую совокупность различных свойств материала, получать достаточно полное представление о свойствах вещества, даже еще не синтезированного. При этом большое внимание важно уделить именно количественному расчету, т. к. явления и процессы, происходящие в открытых системах, определяются большим количеством конкурирующих факторов, не позволяющих ограничиться качественными соображениями.

Окружающий нас мир представим как совокупность открытых систем, включающих в себя атомы, молекулы, кластеры. Самоорганизованная агрегация атомов, молекул, кластеров является естественной в живой природе. В неживой природе целенаправленное создание наноструктурных материалов в режиме «самосборки» требует очень глубокого и достаточно полного представления о взаимодействии атомов, молекул и кластеров. По своим физико-химическим свойствам кластеры занимают промежуточное положение между атомами и молекулами с одной стороны и конденсированным веществом – с другой. Эволюция кластеров ведет к образованию либо газовой фазы, либо конденсированной фазы, проходя ряд метастабильных состояний. Кластеры обладают высокой химической активностью, а цепь процессов перехода от одного состояния к другому является сложной и неравновесной. Еще более сложные процессы взаимодействия кластеров происходят в условиях внешних воздействий, с помощью которых можно и управлять этими процессами [1, 2].

Целью работы является развитие и применение теории несамосопряженных операторов для описания орбитальных возбуждений атомных систем с использованием методов компьютерного моделирования. Практическая реализация этой теории направлена на создание термоэмиссионных, фотоэлектрических и плазменных преобразователей энергии с более высокими значениями КПД по сравнению с КПД современных устройств прямого преобразования тепловой и световой энергии в электрическую. Для оптимизации таких преобразователей часто требуются данные об элементарных процессах, происходящих в веществе [3]. Объектом исследования мы выбрали натрий. В качестве буферных газов – гелий и неон.

В качестве базовой мы использовали теорию функционала электронной плотности, изначально предназначенную для описания основного состояния. Проведено обобщение этой теории для описания одноэлектронных возбуждений, учитывающих ширину энергетических уровней. Идея учитывать ширину энергетических уровней впервые использовалась в [4] при вычислении возбужденных волновых функций в атоме водорода. Описание возбужденных состояний многоэлектронных атомов предложено в [5]. В настоящей работе проведено обобщение этой идеи для описания возбужденных состояний в кластерах.

Нами выполнены самосогласованные расчеты спектров и полных энергий электронов в атомах, двух-, трех- и четырехатомных кластерах натрия, гелия и неона при различных расстояниях между атомами. Задача на собственные значения эффективного одночастичного уравнения Кона-Шема с обменно-корреляционными потенциалами различного вида решалась методом Рутана в базисе гауссовых функций [6]. С помощью численного моделирования показано, что атом натрия, помещенный в поле большой интенсивности, сравнимое с интенсивностью поля в самом атоме, может перейти в возбужденное состояние с полной энергией, большей энергии основного состояния, и пребывать в таком состоянии неограниченно долго. Атом натрия, помещенный в поле большой интенсивности, сравнимое с интенсивностью поля в самом атоме, может перейти в возбужденное состояние с полной энергией, как большей, так и меньшей энергии основного состояния, и пребывать в таком состоянии неограниченно долго. Обнаружены неограниченно долго живущие возбуждения в полях реально достижимых интенсивностей в системе Na-Na при расстояниях между атомами порядка 4 и 11 боровских радиусов, в системе Na-He – при расстояниях между атомами порядка 5.5, 3.3 и 2.5 боровских радиусов, в системе Na-Ne – при расстояниях между атомами порядка 4 боровских радиусов.

Отметим, что использование приближения локальной плотности с различными обменно-корреляционными потенциалами в теории функционала электронной плотности [7-9] приводит лишь к несущественным количественным изменениям. Качественное поведение спектральных линий, полных энергий и межатомные расстояния долгоживущих возбуждений остаются неизменными. Столь необычное вышеописанное поведение атомов натрия в атмосфере гелия и неона вполне объясняется сильными корреляциями ферми систем [10] в поле внешних возбуждений.

Список литературы

1. Слабко В. В., Хачатрян Г. Г., Александровский А. С. Управляемая внешним световым полем самоорганизованная агрегация малых металлических частиц. // Письма в ЖЭТФ, т. 84, с. 360 (2006).
2. Балыкин В. И. Атомно-проекционная параллельная фабрикация наноструктур. // УФН, т. 177, с. 780 (2007).
3. Горбунов Н.А., Flamant G. Качественная модель плазменного фотоэлектрического преобразователя. // ЖТФ, т. 79, с. 72 (2009).
4. Янавичус А., Шучуров В. Водородные волновые функции, учитывающие ширину уровня. // Литовский физический сборник, т. 8, с. 47 (1968).

5. Попов А.В. Решение спектральной задачи для электронов в атоме, учитывающей ширину энергетических уровней. // Оптика и спектроскопия, т. 93, с. 5 (2002).
6. Huzinaga S. Gaussian-type functions for polyatomic systems. I. // J. Chem. Phys., v. 42, p.1293 (1965).
7. Barth von U., Hedin L. A local exchange-correlation potential for the spin-polarized case. // J. Phys. C., v.5, p.1629 (1972).
8. Vosko S. H., Wilk L., Nusair M. Accurate spin-dependent electron liquid correlation energies for local spin density calculation. // Canad. J. Phys., v. 58, p. 1200 (1980).
9. Perdew J. P., Wang L. Accurate and simple analytic representation of the electron-gas correlation energy. // Phys. Rev. B, v. 45, p. 13244 (1992).
10. Шагинян В.Р., Амусья М.Я., Попов К.Г. Универсальное поведение сильнокоррелированных ферми-систем. // УФН, т. 177, с. 585 (2007).

СИСТЕМЫ МОНИТОРИНГА WEB-ПРИЛОЖЕНИЙ

Садвокасов Д.В. – студент
Веревкин М.Н. – к.т.н., директор по развитию ООО «Энтерра-Софт»
Алтайский государственный технический университет (г. Барнаул)

Современные экономические условия предъявляют жесткие требования к качеству программного обеспечения, поэтому, чтобы быть конкурентоспособными предприятиям, занимающимся разработкой и поддержкой приложений, обязательно требуется гарантия постоянной доступности своих приложений в распределенной, гетерогенной и непрерывно меняющейся среде.

Многие современные Web-приложения включают сложный набор распределенных и зависящих друг от друга компонентов, а также обладают динамической структурой, поэтому производительность и степень готовности приложения потенциально зависят от множества внешних факторов. Подобные зависимости практически невозможно свести на нет, либо точно смоделировать в тестовом окружении перед вводом в эксплуатацию. А значит, при работе приложения могут происходить неожиданные ситуации.

В ходе своей работы предприятия сталкиваются со следующими проблемами:

- сложность ИТ-инфраструктуры затрудняет своевременное обнаружение проблем;
- в случае нештатных ситуаций ограничено допустимое время для оперативного поиска и анализ причин сбоев;
- возрастают расходы на обеспечение функционирования ИТ-ресурсов.

В свою очередь, руководство предъявляет к ИТ-службам жесткие требования, согласно которым они должны не только поддерживать работу предприятия, но и обеспечивать новые возможности для реализации целей бизнеса. Среди наиболее актуальных бизнес-задач можно выделить:

- обеспечение непрерывности функционирования бизнес-процессов предприятия;
- оптимизация расходов ИТ-служб, включая оптимизацию использования существующих аппаратных мощностей;
- обеспечение конкурентных технологических преимуществ.

К настоящему моменту около 90% бизнес-процессов коммерческих и государственных предприятий автоматизировано с использованием компьютеров. Информационные технологии сегодня являются абсолютной необходимостью в любом бизнесе и их проблемы способны повлиять не только на жизнеспособность отдельного бизнеса, но даже на национальные экономики. ИТ-сфере уделяется огромное влияние, но программное обеспечение все более усложняется, что приводит к экспоненциальному росту количества связей между компонентами, и в результате надежность программ в целом снижается [1].

Наиболее эффективным средством для решения этих проблем является применение

упреждающих (проактивных) мер – мониторинга, позволяющего перехватывать возникающие сбои и проблемные ситуации на ранней стадии, не допуская их влияния на бизнес-процессы.

В настоящее время существует несколько способов для решения задач мониторинга. Одним из них являются средства отладки и диагностики, поставляемые вместе с прикладной системой.

Несомненным преимуществом такой системы является то, что такой инструментарий есть практически при каждой программной системе и создается он теми же разработчиками. Такие средства хорошо себя проявляют, когда используются с небольшими программами.

При работе с крупными и сложными программными системами они наиболее сильно проявляют свои недостатки:

- неполнота и неадекватность;
- неоднородность и неинтегрируемость частей друг с другом;
- неинтегрируемость со средствами мониторинга системотехники;
- ограниченные возможности представления и экспорта данных [1] .

Преодолеть эти недостатки призваны комплексные системы мониторинга. Комплексные системы мониторинга организуют всесторонний и исчерпывающий мониторинг ИТ-услуг: ИТ-инфраструктуры, связующего и промежуточного ПО, технологических и бизнес-процессов, как взаимосвязного комплекса, обеспечивающего деятельность организации.

Существует два подхода при проектировании комплексных систем мониторинга: фрагментарный и консолидированный. Схематичное представление фрагментированного подхода можно посмотреть на рисунке 1.

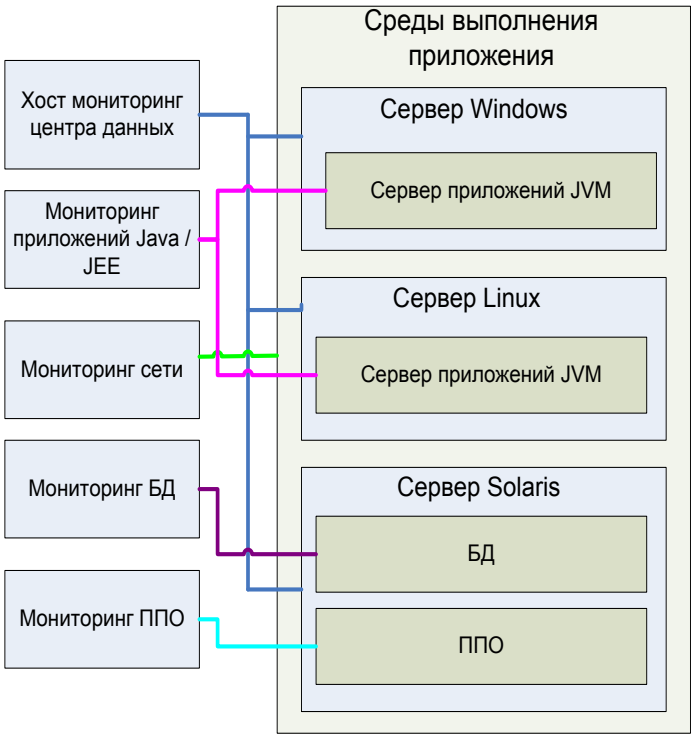


Рис. 1 Фрагментированный подход

При использовании на практике систем мониторинга, построенных на фрагментированном подходе, возникают следующие проблемы:

- Мертвые зоны: некоторые компоненты, от которых зависит приложение, не подвергаются мониторингу, либо данные мониторинга недоступны.
- Черные ящики: само приложение либо один из внешних компонентов не позволяют осуществлять мониторинг своих внутренних процессов.
- Несовместимые или несвязные системы мониторинга: приложение может выполняться внутри крупного вычислительного центра, обращаясь к большому

количеству общих ресурсов, таких как базы данных, высокоскоростные сети хранения данных, а также системы передачи сообщений и различное промежуточное программное обеспечение. Кроме того, компании часто обладают сложной структурой, в которой каждая группа использует собственные системы мониторинга. Если не обеспечить консолидированное представление каждого из внешних компонентов, то каждая группа будет иметь доступ только к небольшому фрагменту общей картины работы приложения.

- Построение отчетов и корреляций постфактум: в попытке решить проблемы, связанные с фрагментированным мониторингом, группа поддержки может периодически запускать процессы сбора данных от различных источников, консолидировать их, а затем строить сводные отчеты. Этот подход часто бывает неэффективен и непрактичен в условиях высокой частоты обновления данных, а отсутствие доступа к консолидированным данным в режиме реального времени снижает быстроту диагностирования проблем. Кроме того, данные, полученные в результате постфактум агрегирования, могут быть недостаточно детальными, из-за чего могут возникать трудности с выявлением важных зависимостей.
- Периодический мониторинг или мониторинг по требованию: существуют средства мониторинга, работа которых требует повышенных затрат ресурсов, поэтому они не могут (или не должны) выполняться постоянно. Таким образом, сбор данных происходит редко, либо только в случае обнаружения проблем. В результате система выполняет лишь минимальный базовый мониторинг и оказывается не в состоянии оповестить вас о проблеме, пока та не приведет к отказу системы, а также может сама ухудшить состояние системы.
- Потеря данных мониторинга: многие средства мониторинга строят полезную картину производительности и степени готовности приложения, однако они либо не настроены, либо просто не поддерживают возможности сохранения данных для анализа за короткие и длительные промежутки времени. Зачастую данные о производительности, представленные вне временного контекста, оказываются малополезными (а то и вовсе бесполезными), так как непонятно, на основании чего можно судить о том, являются показатели хорошими, плохими или критически плохими.

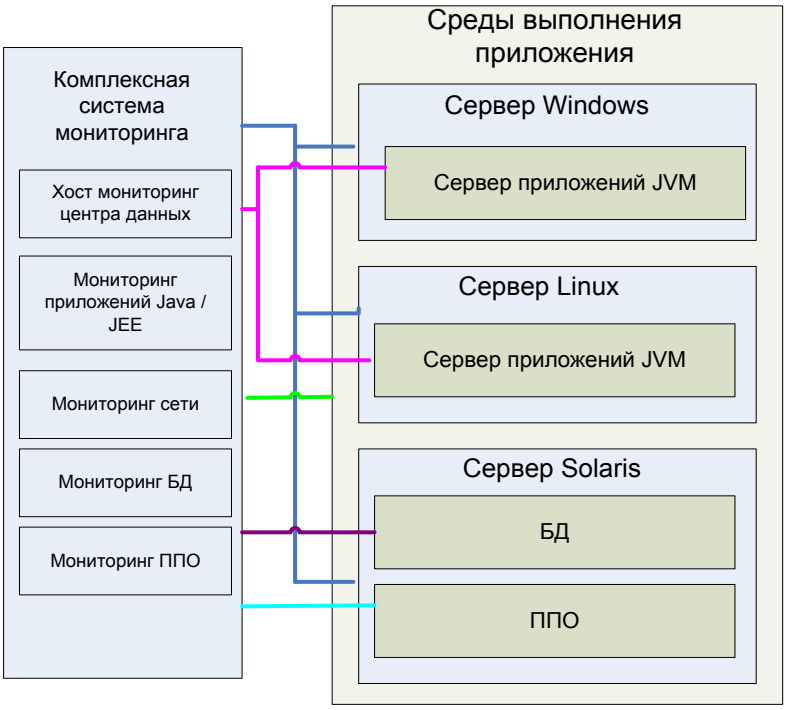


Рис. 2 Консолидированный подход

Консолидированная система не исключает и даже не ставит под сомнение важность узкоспециализированных средств диагностики и мониторинга, низкоуровневые программы для анализа работы сети или решения по управлению центрами данных. Эти средства остаются необходимыми, однако если полагаться только на них, исключив консолидированный мониторинг, то преодолеть проблемы фрагментированного представления будет очень сложно.

Отрицательным принципам, перечисленным выше, можно противопоставить набор следующих свойств, которыми обладает идеальная система:

- Тотальность: система осуществляет мониторинг всех внешних и внутренних компонентов приложения.
- Детальность: система способна производить мониторинг крайне низкоуровневых функций.
- Консолидированность: все собранные показатели проходят через единую систему, строящую консолидированное представление.
- Постоянство: система осуществляет непрерывный мониторинг в режиме 24x7.
- Эффективность: сбор данных о производительности приложения не оказывает пагубного влияния на его быстродействие.
- Работа в режиме реального времени: на основе собранных показателей могут быть построены графики, отчеты, а также выданы предупреждения в режиме реального времени.
- Поддержка временного контекста: собранные показатели помещаются в хранилище данных, что позволяет визуализировать, сравнивать и отображать в отчетах данные за продолжительный промежуток времени [2].

В настоящее время наибольшее распространение получили продукты, которые направлены на составление фрагментированных систем. Каждый программный продукт является самостоятельным и может работать как независимо, так и в комплексе. С помощью этих продуктов можно построить систему с богатым функционалом, включающим в себя выполнение пользовательских сценариев, оповещением в случае сбоев, архивацией данных, ведением журналов и предоставлением подробных отчетов как о состоянии системы в целом, так и каждых ее компонентов по отдельности. Недостатками этих систем является единого центра, где данные могли бы накапливаться для их дальнейшего анализа [3].

В результате проведенной работы, была создана система мониторинга, позволяющая:

- установливать приложения на удаленные HTTP-сервера и сервера приложений;
- вести всесторонний мониторинг ИТ-ресурсов, задействованных в обеспечении бизнес-процессов (СУБД, HTTP-серверы, серверы приложений);
- оперативно фиксировать наличие проблем в функционировании компонентов ИТ-инфраструктуры;
- устанавливать место и характер сбоя;
- определять влияние сбоя в ИТ-ресурсе на предоставление ИТ-сервисов, что позволяет оптимальным образом устанавливать приоритеты в работах по устранению сбоев;
- отслеживать изменения в работе инфраструктуры и предотвращать возможные сбои;
- консолидировать информацию о событиях в едином центре обработки данных;
- удобно и разнообразно предоставлять накопленную информацию мониторинга ИТ-ресурсов с учетом их влияния на бизнес-процессы компании;
- накапливать историческую информацию мониторинга с целью дальнейшего анализа и принятия обоснованных решений
- иметь достаточный набор готовых модулей для мониторинга оборудования и ПО
- предоставлять возможность разработки собственных модулей для мониторинга

Разработанный программный продукт представляет собой консолидированную систему мониторинга, что позволяет решать проблемы фрагментарных систем. Полученный продукт

отличается от аналогов широкими возможностями и более низкой стоимостью.

Список литературы

1. Материалы сайта http://www.jetinfo.ru
2. Материалы сайта http://www.ibm.com/developerworks/ru/library/
3. Материалы сайта http://en.wikipedia.org

СИСТЕМА УПРАВЛЕНИЯ ПРОЕКТАМИ В РАМКАХ СОЦИАЛЬНОЙ СЕТИ НА ПРИМЕРЕ FACEBOOK

Сизов К.Ю. – студент
Веревкин М.Н. – к.т.н., директор по развитию ООО «Энтерра-Софт»
Алтайский государственный технический университет (г. Барнаул)

В связи с бурным развитием сети Internet в последнее десятилетие многие компании (значительно больше, чем за все предыдущие годы), частные лица стали использовать ее ресурсы в своей работе, размещать свои проекты во всемирной сети. Со временем менялись масштабы проектов, расширялась география участников, что вызывало некоторые трудности в организации совместной работы.

На помощь пришли системы управления проектами. Система управления проектами представляет собой набор инструментов, методов, методологий, ресурсов и процедур, используемых для управления проектом [1]. Выполненные в виде веб-приложений, системы управления проектами (далее, СУП) дали возможность успешно руководить деятельностью команд, участники которых могут находиться не только в разных городах, но и в разных странах. Здесь необходимо упомянуть еще об одном аспекте данной работы — о социальных сетях.

Значительное время пользователи сети Internet проводят, общаясь с друзьями, создавая новые знакомства. Активно развивается новый способ взаимодействия людей. Следовательно представляет интерес предоставить инструмент взаимодействия обычным пользователям всемирной сети. Одним из таких инструментов стали социальные сети, которые получили большое распространение в последние годы. Социальные сети — это совокупности участников, объединенных не только средой общения, но и с явно установленными связями между собой [2].

В данной работе представлена система управления проектами, построенная на базе социальной сети.

Система не только использует социальные факторы как основу своего функционирования, так и сама становится дополнительным фактором создания новых сообществ.

Рассмотрим подробнее систему управления проектами.

Прежде всего дадим определение понятию "проект". Проект — это некоторая задача с определенными исходными данными и требуемыми результатами (целями), обуславливающими способ ее решения[3]. Можно дать определение проекта по-другому: Проект – это временное предприятие, предназначенное для создания уникальных продуктов, услуг или результатов [1].

Так что же является проектом? Это создание программного продукта, внедрение новой процедуры или нового процесса на предприятии, осуществление изменений в структуре, кадрах или стиле организации, проведение рекламной акции. Можно приводить сколь угодно много примеров проектов, результатом каждого является законченный продукт или услуга.

Для получения качественных продукта или услуги необходимо управлять процессом их создания. Возникла необходимость в управлении проектом. Это приложение знаний,

навыков, инструментов и методов к операциям проекта для удовлетворения требований, предъявляемых к проекту [1].

Управление проектом помогает разработать план действий и проделать ряд шагов, которые помогут ответить на основные вопросы до непосредственного начала работы. Такие как: что является продуктом? Каковы желания и потребности заказчиков? Кто будет выполнять работу? Сколько времени она займет? Сколько потребуется средств? Какие сбои возможны в работе? Как можно избежать потенциальных проблем? На эти вопросы необходимо ответить заранее, чтобы дальнейшая работа над проектом протекала гладко и эффективно.

Весь необходимый инструментарий для ведения проекта предоставляют СУП.

На сегодняшний день существует огромное число СУП, различающихся как масштабами, функционалом (от мелких, сравнимых с записной книжкой, до систем уровня крупного предприятия), так и используемой платформой (веб-приложения, десктоп-приложения, смешанные и т.п.). Среди прочих можно упомянуть MS Project, OpenAir,Oracle PPM.

Несмотря на значительные отличия друг от друга, большинство СУП имеют на первый взгляд незначительные, но фактически весьма существенные особенности — простому пользователю необходимо привыкнуть к интерфейсу программного продукта, что иногда бывает весьма проблематично из-за неудачного проектирования пользовательского интерфейса; другой немаловажный момент значительно более субъективен и связан с тем, что люди иногда забывают отразить текущие изменения в СУП.

Так каким образом можно минимизировать влияние "человеческого фактора"? Интересным решением является использование факторов социальных — а именно стремление людей к общению. Не секрет, что значительную часть своего рабочего времени многие люди проводят не по назначению, а именно играют в игры, просматривают ресурсы Internet в личных целях, "сидят" в социальных сетях. Однако в связи с тем, что некоторые социальные сети предоставляют независимым разработчикам доступ к своим API для создания на своей платформе всевозможных приложений, возникает очевидное решение выше обозначенной задачи — создание СУП на платформе социальной сети. Что это дает? Теперь каждый раз при посещении социальной сети пользователь сможет проверить статус проектов, менеджер проекта сможет узнавать о присутствии участников в он-лайн, связаться с участниками в личной переписке; использование знакомого окружения снизит затраты на привыкание к интерфейсу приложения.

Другим, гораздо более значимым результатом использования подобной системы является создание новых сообществ. Для работы над конкретным проектом могут объединяться специалисты со всего мира, создавая сильную и сплоченную команду; в результате работы между участниками проекта могут возникать социальные связи, появляться перспективы работы над последующими проектами.

К недостаткам такой системы следует отнести ограниченность возможностей используемой платформы — ограниченность API конкретной социальной сети, что со всеми ошибками платформы, ставит рамки в масштабе создаваемого приложения.

Тем не менее, в результате описанного подхода получится программный продукт, который имеет все шансы стать широко используемым, что вызвано не только высокой степенью интеграции в знакомую среду социальной сети, но и возможностью получить доступ к данным независимо от используемой платформы — будь то ПК или мобильное устройство.

Список литературы

1. Руководство к своду знаний по управлению проектами . Третье издание (Руководство PMBOK®). Project Management Institute, Inc.Four Campus Boulevard , Newtown Square, Pennsylvania 19073-3299 USA / США. Американский национальный стандарт ANSI/PMI 99-001-2004 .

2. Социальные Сети от А до Я. азбука социальных сетей. – http://www.social-networking.ru/ [Электронный ресурс]
3. Мазур И.И., Шапиро В.Д. И др. Управление проектами./И.И.Мазур, В.Д.Шапиро и др./ Справочное пособие. - М.: Высшая школа, 2001 - 875 с.: ил.
4. Мартин П., Тейт К. Управление проектами / Пер с англ. - СПб.: Питер, 2006. - 224 с.: ил.

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УЧЕТА ЭЛЕКТРООБОРУДОВАНИЯ НА ПРИМЕРЕ БАРНАУЛЬСКОГО ФИЛИАЛА ОАО «КУЗБАССЭНЕРГО»

Сикерин А.В. – студент, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В настоящий момент разрозненность информационных ресурсов и ПО, бумажный документооборот определяют низкие функциональные возможности используемых технологических решений, не объединенных в единую технологическую систему, и порождает неунифицированные, затратные процессы планирования, проведения, контроля и обеспечения технического обслуживания и ремонтов электрооборудования. Обилие различных источников информации о состоянии оборудования затрудняет процесс принятия управленческих решений по его эксплуатации и ремонтам. Возникает необходимость в автоматизации существующих производственных процессов по учету, в частности, на предприятиях топливно-энергетического комплекса.

В процессе изучения существующих аналогов были выявлен их принципиальный недостаток – сложность горизонтального и вертикального масштабирования, трудность адаптации к существующим технологическим процессам. Поэтому в процессе проектирования нашей системы были выбраны следующие принципы ее разработки:

- использование общепризнанных и широко используемых стандартов структурирования информации и описания сервисов;
- высокая степень масштабирования программных и аппаратных средств;
- унификация форматов и протоколов межструктурного информационного обмена;
- использование эффективных методов защиты от несанкционированного доступа к информационным ресурсам.
- использование промышленного программного и аппаратного обеспечения;

Основными задачами создания Системы, обеспечивающими достижение поставленных целей, являются:

- обеспечение накопления полученных от специалистов сведений об оборудовании, его истории и особенностях его эксплуатации, а также для систематизации, наглядного представления и централизованного хранения таких сведений;
- обеспечение автоматизации планирования работ по техническому обслуживанию и ремонту оборудования, а также для управления процессом выполнения этих работ;
- обеспечение административного, оперативного и ремонтного персонала оперативной и ретроспективной информацией, необходимой для принятия решений при проведении работ по ТО и ремонту оборудования;
- повышение полноты, точности, оперативности и наглядности такой информации;
- автоматизация получения аналитических отчетов и типовых документов по принятым формам;
- хранение и предоставление данных с целью оценки и прогноза технического состояния оборудования;

- накопление информационной базы, содержащей сведения об оборудовании и его истории, с целью последующего перехода к ремонтам оборудования с учетом его состояния;
- обеспечение точной оценки потребности в материалах и комплектующих и заказа своевременной их поставки, с целью исключить простои в работе оборудования.

Для более детального знакомства с системой рассмотрим некоторые ее особенности. Наибольший интерес представляет проблемы разработки и хранения графической документации, а так же возможности визуального отображения активных зон. Графическая подсистема системы управления предоставляет возможности графического проектирования различных схем оборудования и систем (принципиальных, электрических, функциональных и др.). Изображения, используемые на схемах, размещаются по категориям в библиотеке документов. В системе предусмотрены следующие возможности работы со схемами:

- переход от объекта к его схеме;
- привязка графических объектов схемы к объектам и/или другим схемам
- переход от графического объекта к привязанным объектам базы данных и/или схемам группировка объектов на схеме
- размещение на схеме произвольных текстов
- ввод и отображение комментариев к объектам схемы

На рисунке 1 представлен пример интерфейса редактора активный зон.

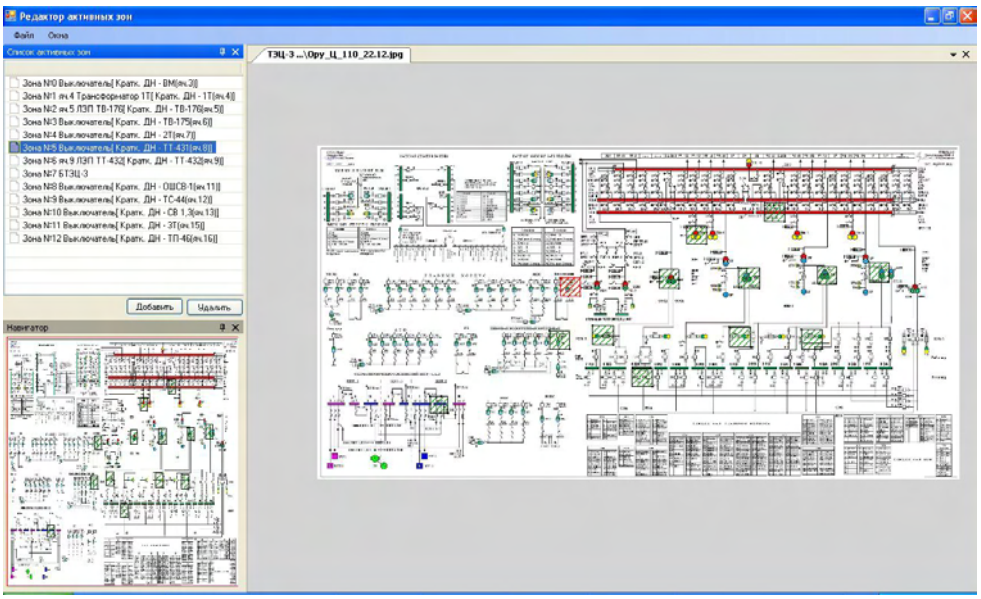


Рисунок 2 Редактор активных зон

Перейдем к рассмотрению системы анализа работоспособности оборудования и технологических систем. Система на основе данных о состоянии оборудования на предприятии позволяет получать информацию о работоспособности любой технологической системы данного предприятия. Визуальное представление типа оборудования и выделение цветом его состояния (в работе, в резерве, в ремонте) позволяет легко оценить текущую ситуацию. Следует отметить возможности по настройке визуального отображения объектов системы. Переход от одной технологической отрасли к другой осуществляется простым изменением отображаемых компонентов оборудования в базе данных.

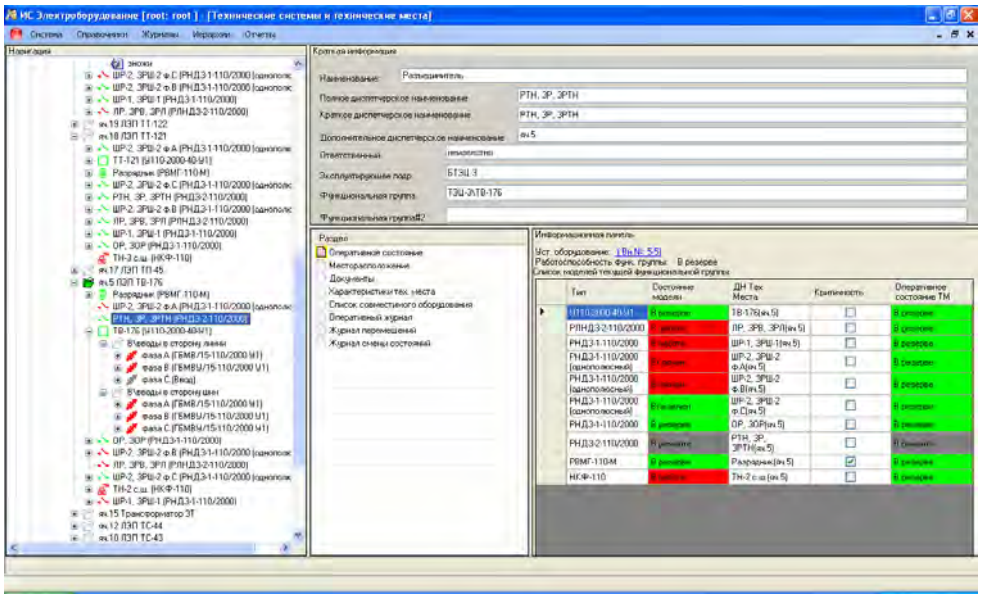


Рисунок 3 Технологические места и технологические системы

На данный момент система внедряется в Барнаульском филиале ОАО «Кузбассэнерго» и хранит информацию:

- о типовом оборудовании электротехническом оборудовании, эксплуатируемом в филиале;
- о технологических местах, на которых эксплуатируется электротехническое оборудование;
- о местоположении эксплуатируемого в филиале электротехнического оборудования;
- о штатной структуре Барнаульского филиала ОАО «Кузбассэнерго»

Опыт внедрения системы показал объем требований, предъявляемых персоналом предприятия к подобным системам, вследствие чего система эволюционирует и развивается. Конечной целью данного проекта ставится получение информационной системы, область применения которой являются предприятия промышленности, энергетики, транспорта, а также любые предприятия и организации, в которых требуется автоматизировать функции технического учета оборудования, хранения технической документации, планирования процессов эксплуатации и ремонта.

РАЗРАБОТКА ИНТЕРНЕТ-СЕРВИСОВ ДЛЯ ОЧИСТКИ ПЕРСОНАЛЬНЫХ ДАННЫХ

Скачков Д. М. – студент, Тамплон А.В. – технический директор ООО «Энтерра-Софт»
Алтайский государственный технический университет (г. Барнаул)

Современные информационные технологии позволяют хранить, обрабатывать и анализировать большие объемы данных. Основная масса данных о клиентах, партнерах, поставщиках современных организаций хранятся в электронном виде, но зачастую эти данные вводятся вручную, или копируются из источников, созданных людьми. Но все люди могут ошибаться, иметь разную подготовку и степень информированности о данных, а также иметь различные социальные и культурные особенности, которые в результате приводят к тому, что большое количество данных имеет низкое качество, или вообще непригодны для дальнейшего использования. В процессе ручного ввода данных возможно появление опечаток, неправдоподобности данных, отсутствия единообразия, внутренних противоречий,

что может обернуться для организации большими потерями. Например, в письме будет неправильно указан адрес, в результате чего оно не дойдет до адресата. Либо будут перепутаны фамилия и имя клиента, поставлен неправильный пол и т. д. Некорректные данные также приводят к неправильным результатам обработки и анализа информации, в результате чего руководством организации могут быть предприняты ошибочные действия. Невозможно даже оценить величину возможных неприятностей, связанных с некорректными данными. Все эти факторы значительно снижают отдачу инвестиций в информационные технологии, в частности в системы управления взаимодействием с клиентами (так называемые CRM-системы), где персональные данные о клиентах являются основными данными, а также препятствуют внедрению информационных систем на предприятиях. В настоящее время на российском рынке информационных систем качеству данных не уделяется должного внимания, а между тем на Западе качество данных уже давно признано приоритетным аспектом разработки информационных систем.

Целью работы является создание решения, предоставляющего следующие возможности очистки персональных данных:

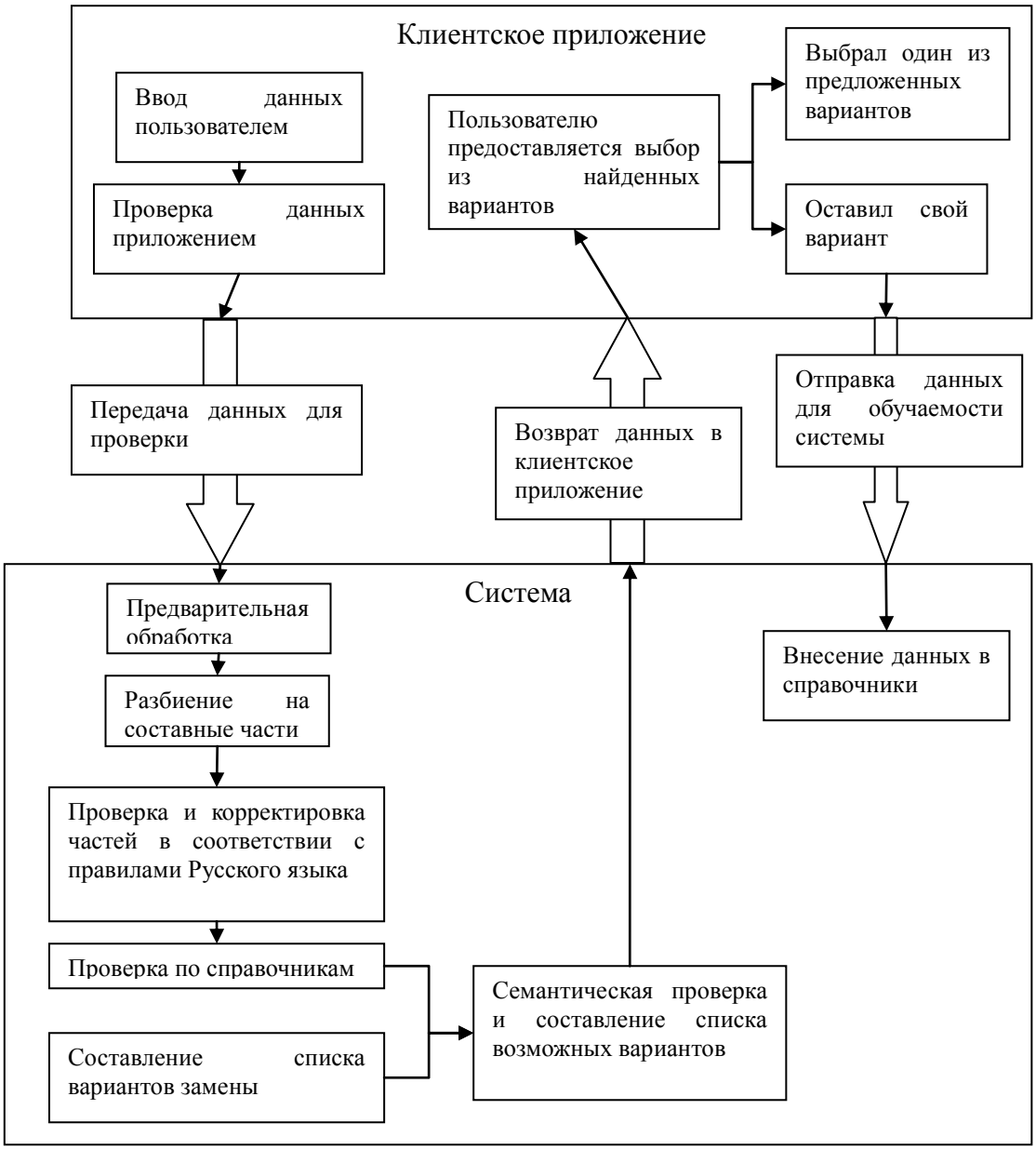
1. Устранять неполноту данных. Например, если указан телефон, то определять код города; если указан населенный пункт, то определять край или область;
2. Устранять некорректность данных. Например, исправлять пол, если он не совпадает с ФИО; если в поле адрес стоит телефон, и наоборот, менять их местами; если данные написаны транслитом, то переводить их на Русский язык; устранять опечатки; проверять адреса по по справочнику КЛАДР и т.д.
3. Разделять смешанные данные. Например, разделять ФИО, Адрес и индекс населенного пункта, адрес и телефон.
4. Приводить данные к единообразному виду. Например, телефон к виду «+7-913-925-1556» ; адрес к виду «399774 Липецкая обл, Елецкий р-н, г Елец, ул К.Маркса»; дату к виду «03/02/2004»; БаРнАуЛ к Барнаул;
5. Определять, являются ли одни данные дубликатами других данных. Например записи «город Елец, ул Маркса, 9» и «399774 Липецкая обл, Елецкий р-н, г Елец, ул К.Маркса, д. 9» являются адресами одного и того же дома.

Сервис должен обладать следующими функциональными особенностями:

1. Обучаемость, состоящая в том, что система может добавлять слово в справочник, если это слово было использовано несколько раз. При повторении ошибки (и последующем исправлении ее пользователем), система впоследствии сама может принять решение об исправлении ошибки (либо помещать вариант как более вероятный в верхушке списка возможных исправлений).
2. Возможность использования стандартных справочников (КЛАДР, справочник телефонных кодов и т.д.), а также возможность подключать дополнительные справочники.
3. Возможность выдавать не только единственный результат, но и группу наиболее вероятных вариантов.
4. Производить проверку как по отдельным элементам (ФИО, Телефон, Адрес) так и по целой записи. В случае целой записи система должна уметь классифицировать предоставленные ей данные.
5. Собирать статистику по работе системы и предоставлять инструменты по ее анализу

На начальном этапе разработки системы самой важнейшей задачей является изучение теоретической информации и проектирование основных модулей системы.

Схема обработки данных системой:



На последующих этапах разработки планируется произвести объектно-ориентированный анализ системы и проектирование основных элементов системы.

Список литературы

1. В.Э. Карпов. Об одной задаче очистки и синхронизации данных [Электронный ресурс] / В.Э. Карпов, И.П. Карпова // Информационные технологии. – 2002. – № 9. – Режим доступа: http://www.raai.org/about/persons/karpov/pages/dscrubb/dscrubb.html

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ДЛЯ ТЕСТИРОВАНИЯ ПРОГРАММ

Старолетов С.М. – аспирант, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Целью данного теоретико-практического исследования является создание своей математической модели современной программы (распределённой недетерминированной программной системы) и применение её при проведении тестировании сложных систем при помощи разрабатываемых средств. Актуальность работы вытекает из необходимости применения различных методик при тестировании программ, перспективности применения моделей в тестировании программ, необходимости повышения культуры тестирования, отсутствия на рынке средств тестирования при помощи построения моделей, понятных для среднестатистического тестировщика без математического образования.

Настоящее исследование проводится в сфере тестирования программного обеспечения, важной части процесса разработки ПО, которая отвечает за выявление ошибок (дефектов) в программном обеспечении.

Математик Дейкстра однажды сказал: “Тестирование программ может использоваться для демонстрации наличия ошибок, но оно никогда не покажет их отсутствие” [6].

Определение: Проблема тестирования заключается в том, возможно ли для любой программы определить, выполняет ли она поставленный алгоритм или имеет ошибки и не выполняет поставленный алгоритм?

Формально, допустим, если программа определяется как набор состояний машины Тьюринга и перед каждым шагом по переходу в следующее состояние осуществляется вывод значений переменных, и если после завершения программы вывод окажется в точности равен ожидаемому, то программа признается верной. Проблема тестирования состоит в том, возможно ли такую проверку сделать для любой программы.

Теорема: Проблема тестирования неразрешима.

Доказательство: Методом представления программы в виде машины Тьюринга и сведения к NP-полным проблемам переводимости или остановки.

Доказанная теорема говорит о том, что невозможно в общем случае доказать правильность любой программы, имея только её описание. Возникает вопрос — как же тогда осуществляется проверка правильности программ сегодняшними методами? В настоящее время обычно доказывается правильность специальных видов программ (алгоритмов) с точки зрения какого-то из методов, при этом программа может оказаться правильной с точки зрения данного метода, но неправильной с точки зрения другого. Обычно же доказывается не правильность программы, а неправильность ее, т.е. ищется какая-либо ошибка в программе. Если таких ошибок не найдено, программа считается правильной до выявления ошибки пользователями в процессе ее эксплуатации.

Данная работа посвящена выработке стратегий тестирования программ, которые помогут разработчикам находить ошибки в взаимодействующих компьютерных системах, используя специально построенную модель для таких систем. Наличие кроме самой программы её модели позволит использовать для проверки правильности дополнительный инструмент, который позволит сравнительно легко находить ошибки и узкие места в программах, анализирование которых без использования моделей было бы очень сложной задачей.

О важности подхода говорит, например, то, что исследовательское подразделение компании Microsoft исследует MBT (Model-based testing – тестирование на основе моделей) с 1999г., используя свою, отличную от предлагаемой далее, модель [7] и применяет результаты исследований для внутреннего использования.

Методы исследования включают анализ взаимодействующих систем, формализация понятий, применение теории конечных автоматов, теории графов, теории вероятности и Марковских случайных процессов, теории формальных языков, теории алгоритмов.

Ядром исследования является математическая модель взаимодействующих систем. Система представляет собой набор компонентов и связей между ними, а также глобальные

множества отсылаемых сообщений и общих блокируемых ресурсов:

$$M = (A^*, CP(A^*), Msg, Res)$$

Здесь A^* - множество расширенных вероятностных многопоточных конечных автоматов, моделирующих поведение компонентов системы в виде мультиагентных программ (модель по сравнению с [] была существенно расширена):

$$A = (q_0, Q, F, \delta, \gamma, E, msg \subseteq Msg, res \subseteq Res),$$

где Q - множество состояний, $q_0 \in Q$ - начальное, $F \subseteq Q$ - множество заключительных состояний; E - множество событий и исключительных ситуаций; msg – подмножество глобального множества отсылаемых сообщений; res – подмножество глобального множества общих ресурсов; логика переходов описывается недетерминированной функцией переходов "по вершинам" δ и функцией переходов "по дугам" γ.

$CP(A^*)$ определяет множество точек сопряжения компонентов системы, а также мощности их связей (*1:1*,*1:N*, по принципу "рукопожатия").

Определение: состояние тестируемой системы — это участок исходного кода программы (либо компонента программы), который разработчик или архитектор системы определяют как единый логический результат деятельности фрагмента программы.

Мы будем определять логическое состояние как набор строк исходного кода программной системы. Вообще, реальная распределенная система может состоять из нескольких компонентов, каждый компонент представляет собой проект (в интегрированной среде разработки проект это - единая сущность, связывающая относящиеся к нему файлы с исходными кодами и средствами для настройки компилятора для создания объектного кода из набора исходных файлов):

$$System = \cup Project$$

(здесь используется операция объединения множества проектов как неупорядоченная их совокупность).

Каждый проект, в свою очередь, состоит из нескольких файлов с исходным кодом

$$Project = \cup SrcFile$$

на каком-то языке программирования, а каждый файл с исходным кодом — из набора строк с кодом программы некоторой длины:

$$SrcFile = \bigcup_{i=1\ldots|SrcFile|} S_i .$$

Обобщив сказанное, можно сказать, что

$$Project = \bigcup_{\substack{f \subset SrcFile \\ i=1\ldots|f|}} S_{fi} .$$

Тогда состояние $q \in Q$ относительно к коду программы определим так:

$$q = \left\{ \bigcup_{i=p\ldots r} S_{fi} \middle| f \in SrcFile, в\ файле\ f\ с\ линии\ p\ до\ линии\ r\ код - логически\ значимый \right\}$$

Значимость кода состояния для работы системы определяется разработчиками.

Отметим, что состояния не могут перекрываться, т.е. если q_1 и q_2 - состояния программы ($q_1 \in Q$, $q_2 \in Q$), в файле f и

$$q_1 = \bigcup_{i=p_1\ldots r_1} S_{f_i} \text{ и } q_2 = \bigcup_{i=p_2\ldots r_2} S_{f_i} \text{, то } \left(\bigcup_{i=p_1\ldots r_1} S_{f_i} \right) \cap \left(\bigcup_{i=p_2\ldots r_2} S_{f_i} \right) = \varnothing .$$

Определение: Переход из одного состояния в программе определяется двумя позициями в ее исходном коде: во-первых, это место в программе, в котором осуществляется какое-либо взаимодействие, с точки зрения разработчика, приводящее к изменению состояния системы (начало перехода), а во-вторых, это место начала состояния, в которое осуществляется переход (конец перехода).

Формально, переход T в программе из состояния q_x в состояние q_y определяется строкой начала перехода и состояния, в которое производится переход:

$$q_x \to q_y : Transition_{xy} = \{ (u, q_y) | q_x = (\cup_{i=p_x,\dots r_x} S_{fi}),$$
$$f \in SrcFiles, p_x \leqslant u \leqslant r_x, S_{fu} \text{ определяет взаимодействие} \}$$

То, что именно определяет взаимодействие, как и при описании состояния, решается разработчиком или архитектором системы.

Логика работы по переходу определяется с помощью функций переходов:

$$\delta : Q \times D \times P \times N \times T \times Msg \times Res \longrightarrow$$
$$2\text{^}(((T \times Q)^* \cup Q) \times Msg^* \times Res)$$

В соответствии с δ, находясь в состоянии из Q кратности N по действию из D с вероятностью из P в потоке(thread) из T и по полученному сообщению из Msg, после установки блокировки ресурса на Res, модель системы недетерминировано переходит либо в несколько состояний, создав несколько новых потоков из T, или просто в следующее состояние в текущем потоке; отсылая сообщения из Msg и разблокируя ресурс из Res.

В момент перехода в соответствии с δ, предлагаемый расширенный автомат может сообщить еще переходы в соответствии с γ:

$$\gamma : Q \times Q \to E^*$$

Это означает, что при переходе из состояния в состояние возможно возникновение и обработка конечного числа событий или исключительных ситуаций из E (event,exception).

Рассмотрим поподробнее основные используемые множества применительно к реальным процессам в программировании.

D - множество действий, событий, по которому происходит переход в состояние.

Поскольку слово «событие» имеет отношение к работе программы и далее будет использоваться в другом контексте, поэтому предлагается называть «действием» некоторое словесное условие для перехода между состояниями. В графической интерпретации автомата действие — это надпись на дуге между двумя состояниями. В определении стандартного конечного автомата обычно присутствует Σ - входной алфавит (автомат, читая символы входного алфавита, переходит в состояния). Мы же полагаем $\Sigma = D$, и далее понятие «входной алфавит» опускаем, ведь наш автомат предназначен не для распознавания, а для моделирования.

$P \subseteq \mathbb{R}$ – вероятность перехода. Для анализа правильности сложных систем целесообразно добавить в функцию переходов элемент вероятности. После этого наш автомат будет считаться вероятностным конечным автоматом [1], и мы будем применять такой автомат для моделирования сложной взаимодействующей системы, где элемент случайности — это переход в следующее состояние. Отметим, что таким образом предлагается дистанцироваться от логики работы программы в виде контроля пред- и пост- условий на переменные системы, как это делается при верификации систем в [7]. Облегчая для пользователей, которые создают модель системы, ее описание, мы будем использовать вероятностные переходы для моделирования возможной сложности работы программы.

Например, для моделирования *if (A) B; else C*, можно построить систему предусловий (для выполнения B необходимо, чтобы A было истинно, а для выполнения C — чтобы A было ложно), но мы поставим себе задачу прохождения ветви B с условной вероятностью $P(B|A)$ и ветви C с вероятностью $P(C|A)$.

Вероятность перехода определяется для каждой из пар состояний (состояние из которого осуществляется переход; состояние, в которое осуществляется переход на следующем шаге). Здесь можно говорить о матрице вероятности переходов, P_{ij}, где $1 \le i, j \le |Q|$.

Естественно предполагать также, что:

$$0 \le P_{ij} \le 1,$$

$$\forall i : 1 \le i \le |Q| \sum_{j=1\ldots|Q|} P_{ij} = 1$$

(Матрица P_{ij} является стохастической матрицей)

Возникает вопрос, откуда брать вероятности перехода из состояния в состояние? Ответ на этот вопрос могут дать предварительно как разработчики системы, так и данные о переходах между состояниями после реального запуска и отработки системы.

Определение: априорными вероятностями системы назовем вероятности переходов между состояниями вероятностного конечного автомата, которые заданы разработчиками/архитекторами системы исходя из предположений о работе системы. Априорные вероятности определяют ожидаемое поведение системы.

Определение: апостериорными вероятностями системы назовем вероятности переходов модели, каким-то образом вычисленные после работы программы на основании статистики переходов. Апостериорные вероятности определяют реальное поведение системы после её запуска.

Можно отметить, что апостериорные (вычисленные) вероятности могут быть использованы как априорные при следующем запуске моделирования.

Здесь мы используем то, что апостериорные вероятности каким-то образом вычисляются в процессе работы системы, т.е. обеспечивается связь и от модели к системе и от системы к модели. Организация такого взаимодействия будет рассмотрена далее.

Заметим, что мы осуществляем переход из текущего состояния в следующее без учета предыдущих посещённых состояний. Тогда поведение нашей модели мы можем описать как марковский случайный процесс с дискретными состояниями и дискретным временем [2] (цепь Маркова). Используя априорные вероятности, можно, с помощью аппарата марковских процессов, установить вероятности, того что на каком-то шаге модель будет находиться в каком-то состоянии, установить условия наличия финальных (установившихся) состояний и их значения.

E – множество событий или исключений. Можно отметить, что современные программы построены не на линейном вычислительном процессе, который представляет собой просто переходы из состояния в состояние, а являются событийно-ориентированными [3].

Все современные языки программирования имеют механизм обработки исключительных ситуаций, обычно возникающих в программе в результате ошибок времени выполнения и позволяющих без потерь для работы программы корректно обработать ошибку и продолжать выполнение, при условии, что разработчик намеренно предусмотрел данную ситуацию. Исключение и событие похожи, поскольку исключение — это событие в ответ на ошибку.

Для симуляции работы событийной системы множество состояний разделим на те состояния, которые достижимы в программе в результате её последовательной работы (Q_0), и на те, которые достижимы как обработчики событий или исключений (Q_E): $Q = Q_0 \cup Q_E$.

$E^* = \varnothing \times E \times \ldots \times E$ - итерация E;

$E = Q_E \times P_E \times W$ - множество, описывающие событие или исключение (Event or Exception). Событие или исключение определяется:

- состоянием из множества состояний-обработчиков исключений Q_E - тем состоянием, куда будет осуществлен переход;
- вероятностью того, что сработает именно данный обработчик (обозначим такую

вероятность с индексом «E»: $p_E \in P_E$);

- а также флагом типа boolean W ($W=\{true,false\}$), показывающим то, что переход к обработке данного события определяет переход в так называемое ошибочное состояние.

Определение. Ошибочным состоянием системы назовем такое состояние, попадание в которое говорит о возникновении какой-либо ошибки в программе с точки зрения разработчика.

Частое нахождение системы в таких состояниях говорит о ненадежности окружающей среды программы. С другой стороны, поскольку ошибочное состояние может объявляется у обработчика событий или исключений (скорее у исключений), то описание такого состояния уже само по себе является указанием со стороны разработчика на возможность ошибочной ситуации.

Графическое представление перехода с событиями показано на рисунке 1. Осуществляется вероятностный переход в состояния k и j, при этом возможно возникновение в момент одного из переходов событий (исключений) $E_1 \ldots E_M$.

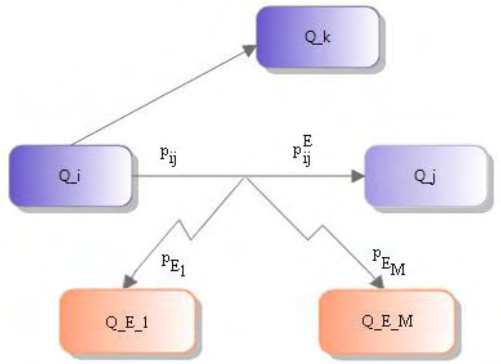


Рисунок 1 – Переходы и события

Предположим, что происходит переход из состояния q_i в, q_j вероятность перехода при этом p_{ij} (теперь мы можем говорить о вероятности корректного «выхода» из состояния q_i (покидание данного состояния), но она уже не будет соответствовать вероятности «выхода из состояния q_i и захода в состояние q_j»); кроме того, возможно возникновение k исключительных ситуаций $E^k=E_1 \times \ldots \times E_k$ или событий с вероятностями их возникновения при данном переходе p_E, m=1..k.

Тогда вероятность того, что система перейдет из состояния q_i в q_j, скорректированная с учетом событий: $p_{ij}^E = 1 - \sum_{m=1..k} p_{ij} \cdot p_{E_m}$.

Данное соотношение следует из формулы произведения вероятностей, поскольку для наступления события q_E необходимо предварительное срабатывание перехода с вероятностью p_{ij}.

Далее мы рассмотрим моделирование взаимодействующих систем.

Распределенной системой назовем систему, в которой работают параллельные процессы или потоки. Строго говоря, процессы или потоки имеют отличия в общности данных (потоки имеют дело с общими данными программы, а процессы-нет, они взаимодействуют только с разделяемыми данными со стороны ОС). Поскольку мы моделируем взаимодействие, а не анализ данных, то для нас не имеет принципиального значения — процесс или поток, главное в том, что в системе выполняются параллельные действия.

Определение. Моделью потока (или нити, англ. thread) в системе назовем параллельно выполняющиеся действия, заданные своей логикой работы в виде подавтомата; действия

выполняются параллельно действиям потока, который создал данный поток (поток-родитель) и всем его другим дочерним потокам, действиями являются переходы согласно функции переходов, в том числе сложные, и создание других потоков.

Поток определяется:

- Потоком-родителем, создавший данный поток (T_{parent}). Родитель есть у всех потоков, создаваемых в системе, кроме главного потока приложения.
- Подавтоматом, который отвечает за логику работу потока (A'). Подавтомат определен на группе состояний $Q' \in Q$, и эти состояния и взаимодействие между ними определяют работу потока;
- Именем потока (строкой).

Для создания потока необходимо выполнить особое действие по переходу из состояния в несколько состояний, создав при этом несколько потоков. Предлагается называть такое действие «вилкой» или операцией «fork»(англ. вилка), по аналогии с популярной системной функцией созданием копии процесса в UNIX системах (строго говоря мы поступаем совсем не так, как одноименная функция - не создаем копию процесса, а создаем новый поток, находясь в каком-либо состоянии). Таким образом, функция переходов была дополнена переходом с созданием потоков: $\delta \supseteq fork\colon Q \to (Q \times T)^*$ (Рисунок 2).

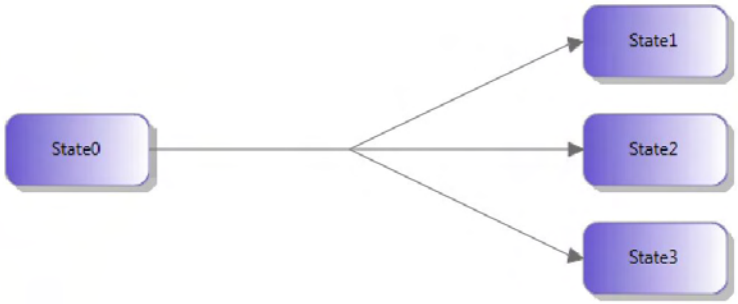


Рисунок 2 – Создание потоков.

Рассмотрим следующую ситуацию: создание потока в цикле (Рисунок 3):

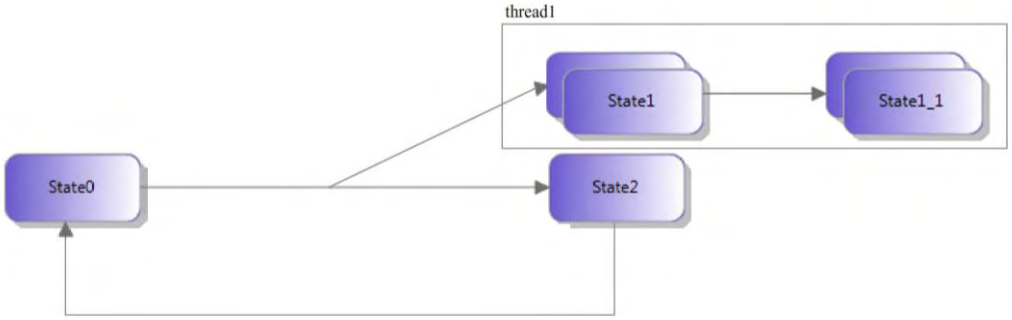


Рисунок 3 - Создание потока в цикле.

В точке создается поток, и главный поток продолжает работу, и через конечное число состояний создает такой же поток. Это приводит нас к определению кратности состояния.

Определение. Кратностью состояния назовем целое число N, которое показывает, что в данном состоянии может находиться одновременно не более N потоков. По умолчанию кратность состояния принимается равной 1.

Кратность потока – это кратность его первого состояния.

Вместе с понятием кратностей состояния и потока предлагается ввести и понятие перехода по кратности. Можно заметить, что на рисунке 3 состояния State1 и State1_1

является кратным, однако вечно создавать потоки невозможно, поскольку их число в операционной системе ограничено, и надо промоделировать выход из операции создания потоков. Другой вариант перехода по достигнутой кратности, который можно привести в пример, это пример чата, когда сервер ждет подключения минимум двух клиентов (в модели — создания потока кратности 2, инкапсулирующего поведение клиента), чтобы был разрешен обмен сообщениями между ними.

Определение. Переход по кратности N осуществляется, либо когда кратность текущего потока равна N : $N(T_{current})=N$, либо когда кратность заданного потока T с точки зрения текущего потока равна N : $N(T,T_{current})=N$.

Средства моделирования синхронизации потоков:

- сообщения;
- блокировка ресурсов.

Обычно клиент-серверные приложения взаимодействуют через отсылку и получение сообщений, при этом некоторые языки программирования построены полностью на обмене ими.

Определение. Сообщение в модели — это элемент множества Msg, характеризуется следующими атрибутами:

- Состояние, из которого сообщение отсылается ($q_{from} \in Q$);
- Состояние, в которое сообщение посылается ($q_{to} \in Q$);
- Идентификатор сообщения $msgrid \in MsgID$. Идентификатор определяет сообщение;
- Тип доставки сообщения, определяет, как сообщение будет доставляться в кратные состояния ($type=\{'\forall','N'=n, '\forall/me'\}$):
  - Сообщение «всем» («broadcasting») - отсылается всем экземплярам потока в кратном состоянии;
  - Сообщение конкретному экземпляру потока с его номером;
  - Сообщение «всем, кроме меня» - отсылается всем экземплярам потока из потока этого же экземпляра, кроме самого себя.
- Строка-текст сообщения (необязательно), $txt \in String$.

Таким образом, $msg=(q_{from}, q_{to}, msgid, type, txt)$.

Сообщения — это не обязательно клиент-серверный обмен с использованием, например, сетевых сокетов, но и оповещение одного объекта системы об изменении своего состояния для других её объектов.

Отсылка и получение сообщений показана на рисунке 4.

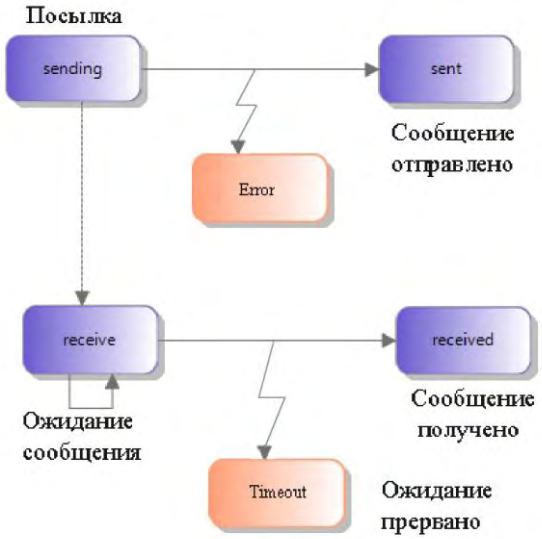


Рисунок 4 – Отправка и получение сообщений.

Системы, запрограммированные многопоточно, имеют обычно, проблемы синхронизации потоков. В одном случае это изменение общих данных одним потоком, когда другой считает эти данные не измененными, в другом случае — это проблемы их синхронизации после выполнения подзадач большой задачи в породившем их потоке.

Данные проблемы возможно решить вводом в модель некоторых общих ресурсов, получить доступ к которым можно, только если этот ресурс не заблокирован другим потоком. Блокируя ресурс потоком, можно выполнять некоторые действия, которые другой, конкурирующий поток не должен выполнить одновременно, и он будет ждать освобождения данного ресурса; после выполнения действий поток, который захватил ресурс, должен его освободить и дать другим потокам возможность выполнить действия, которые нельзя выполнять одновременно (например, доступ к файлам или глобальным данным).

Во всех современных операционных системах присутствуют примитивы синхронизации, такие как семафор или мьютекс, которые блокируются или разблокируются при помощи атомарных системных вызовов в ОС.

Имея модель программной системы для целей ее тестирования, тестирующая среда выступает внешней по отношению к модели, так же как и операционная система выступает по отношению к программе. Соответственно, внешняя тестирующая среда также может предоставлять модели общие блокируемые ресурсы и контролировать их использование.

Мы будем моделировать синхронизацию потоков аналогично работе критической секции synchronized в Java и lock в C#.

В определение расширенного автомата и модели системы было добавлено множество общих блокируемых ресурсов *Res*:

А также добавим операции *block*(*R*), $R \in Res$ и *unblock*(*R*), $R \in Res$, блокирующие и разблокирующие ресурсы из *Res*.

Операция *block*(*R*) проверяет, кто на текущий момент является хозяином ресурса *R* (*Host*(*R*)):

Если $Host(R)=T_{current}$, не делается ничего;

Если $Host(R)=\varnothing$, то устанавливается $Host(R)=T_{current}$ и блокировка считается успешной, производится переход по успешной блокировке в соответствии с расширенной функции переходов;

Если же $Host(R)\neq\varnothing$, $Host(R)=T$, $T\neq T_{current}$, то $T_{current}$ блокируется (т.е. осуществляется переход $q \to q$, $q \in Q'$, $T_{current}$: Q' и последующая проверка *Host*(*R*)). Блокировка может быть прервана по тайм-ауту, причем переход в состояния между блокировками запрещен.

Операция *unblock*(*R*) делает следующее:

Если $Host(R)=T_{current}$, то $Host(R)=\varnothing$ и блокировка снимается;

Если $Host(R)\neq\varnothing$ и $Host(R)\neq T_{current}$, то ничего не делается.

Таким образом, меняется кратность состояний внутри потока между операциями Block и UnBlock и становится равной 1 (Рисунок 5).

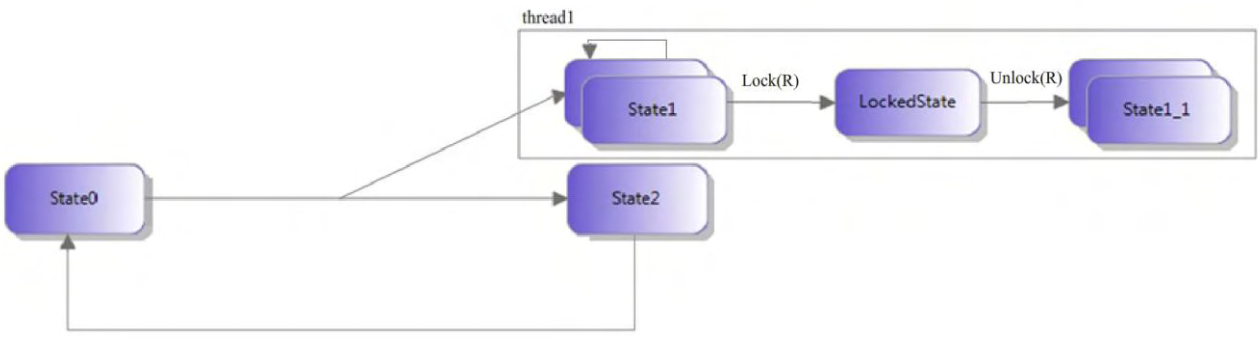


Рисунок 5 - Блокировки, разблокировки и кратные состояния.

Рассмотрим теперь точки сопряжения между компонентами системы. Можно рассматривать связи в качестве модели рукопожатия (Рисунок 6):

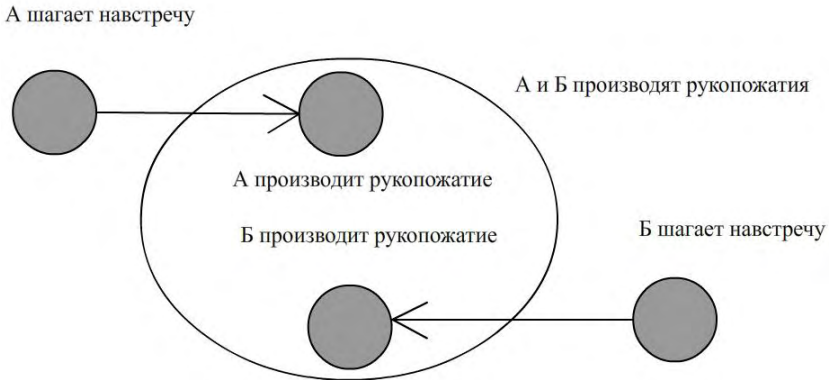


Рисунок 6 — Рукопожатие.

Два человека делают шаг друг к другу и осуществляют рукопожатие.

А и Б в момент рукопожатия находятся каждый в своем состоянии, а также в общем, или состояние рукопожатия А соответствует состоянию рукопожатия Б.

Таким образом, в нашей модели, когда один из компонентов системы находится в каком-то состоянии, другой в это же время может находиться в некотором количестве состояний; когда он перейдет в другое состояние, связанный компонент перейдет в другой набор состояний. Таким образом, множество $CP(A_1, A_2)$ в общем случае задает точки сопряжения между парой компонентов, так что

$$CP(A_1, A_2) = \{(Q_1,Q_2) \mid Q_1 \subseteq Q(A_1), Q_2 \subseteq Q(A_2)\}.$$

В общем случае, некоторому количеству состояний автомата A_1 соответствует некоторое число состояний из A_2. По аналогии из теории реляционных баз данных — как связи между таблицами, у нас так определяется связь мощности *M:N* между двумя компонентами системы.

На практике, однако, связи *M:N* не используются, поскольку таким образом трудно определить связи между конкретными состояниями. Мы будем предполагать, что взаимодействие между компонентами модели описывается точками сопряжения мощностями *1:N* или *1:1*:

$CP(A_1, A_2) = \{(q_1,Q_2) \mid q_1 \subseteq Q(A_1), Q_2 \subseteq Q(A_2)\}$, связь 1:N — одному состоянию автомата A_1 соответствует несколько состояний A_2;

$CP(A_1, A_2) = \{(q_1,q_2) \mid q_1 \subseteq Q(A_1), q_2 \subseteq Q(A_2)\}$, связь 1:1 – одному состоянию A_1 соответствует одно состояние из A_2.

Следующим шагом исследования является язык описания предложенной модели, который не зависит от используемых при создании системы языков программирования (компоненты распределенной системы могут быть написаны на различных языках) и определяет формальное текстовое описание разработанной модели.

Применяется принцип ”код и модель-одно целое”, при котором описание модели на языке встраивается в исходный код системы в виде комментариев специального вида. Сущности “состояние”, “переход”, “поток”, “исключение” и др. описываются на разработанном языке XML тегами с атрибутами.

Далее, для популярных сред разработки Eclipse и Microsoft Visual Studio разрабатываются расширения (Plugin, VS Package) для визуального описания моделей прямо при создании системы (на одно из расширений получено авторское свидетельство). На рисунке 7 показано расширение для Visual Studio .NET 2008

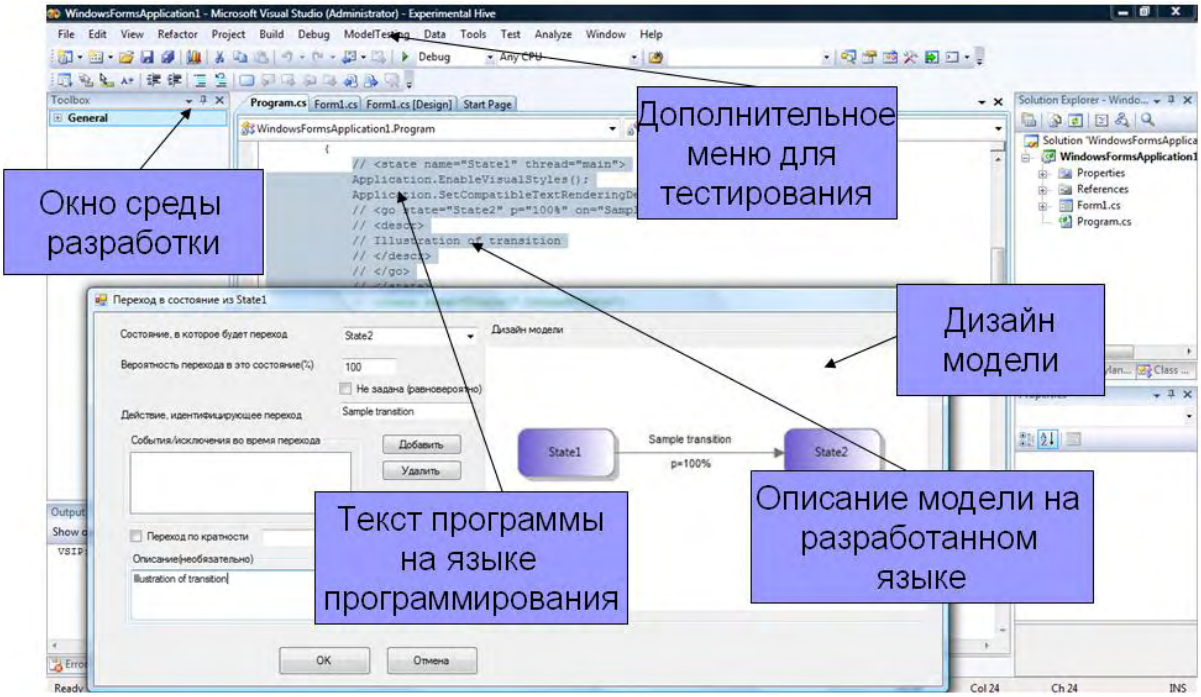


Рисунок 7 – Расширение IDE.

Собственно тестирование заключается, во-первых, в анализе модели без запуска системы (off-line), когда сложная структура модели специальным методом преобразуется в ориентированный граф и далее исследуется, и, во-вторых, с запуском системы и сравнение реальной работы системы и модели с использованием сервера тестирования (on-line, код обращения к серверу встраивается препроцессором в места описания модели).

Критерии корректности модели и системы:

А) Формальные:

- Описаны все состояния, что используются в переходах;
- не существует переходов в несуществующие состояния;
- нет состояний, в которые ничего не идет;
- матрица переходных вероятностей стохастическая;
- состояния не перекрываются;
- у сообщений есть отправитель и получатель;
- компоненты системы связаны через точки сопряжения.

Б) Тестируемые:

- Переход из состояние в состояние и обработка событий происходит строго по модели;
- кратность состояний не превышается;
- связность через точки сопряжения не нарушается;
- все сообщения доходят; блокировки срабатывают.

Таким образом, в данной статье обоснована и предложена модель взаимодействующей системы и применение ее в целях тестирования распределенного программного обеспечения.

Разработка поддержана грантом “УМНИК”. Ведется реализация по утвержденному плану и смете.

Список литературы

1. Бухараев Р. Г. Основы теории вероятностных автоматов/ Бухарев Р.Г. М.: Наука, 1985
2. Вентцель Е.С. Теория случайных процессов и ее инженерные приложения / Вентцель Е.С., Овчаров Л.А. М: Наука. Гл. ред. физ.-мат. лит. - 1991 - 384 с.

3. Иртегов Д.В. Событийно-ориентированные архитектуры. Программирование с использованием POSIX thread library. / Иртегов Д.В. Слайды. Режим доступа - http://edu.dgu.ru/Res/SUN/Irtegov_Tech/Lecture_9_Event_driven_architectures_slides.pdf
4. Старолетов С.М. Мультиагентная модель распределенной системы для проведения model-based testing современного ПО / Старолетов С.М. Технологии Microsoft в теории и практике программирования: Тр. VI Всерос. конф. студентов, аспирантов и молодых учёных. Центральный регион. Москва, 1-2 апреля 2009г. – М: Вузовская книга, 2009.- с. 37-38.
5. Старолетов С.М. Тестирование распределенных приложений на основе построения моделей/Старолетов С.М., Крючкова Е.Н.//Прикладная информатика – М: Market DS publishing. – 2008. – N6. – С.124-134.
6. Тестирование программного обеспечения [Электронный ресурс] / Режим доступа- ru.wikipedia.org/wiki/Тестирование_программного_обеспечения
7. Andreas Blass, Yuri Gurevich, Lev Nachmanson, Margus Veanes. Play to Test. Режим доступа – http://research.microsoft.com

РАЗРАБОТКА WEB-ИНТЕРФЕЙСА ДЛЯ ДОСТУПА К ЭЛЕКТРОННОЙ БИБЛИОТЕКЕ ОБРАЗОВАТЕЛЬНЫХ РЕСУРСОВ АЛТГТУ НА ОСНОВЕ СЕРВЕРА Z39.50

Теряев Р.А. – студент, Лукоянычев В.Г. – к.т.н., доцент
Алтайский государственный технический университет (г. Барнаул)

В современный век информационных технологий все больше организаций внедряют информационные системы. Они предоставляют практически неограниченные возможности размещения, хранения, обработки и доставки информации любого объема и содержания на любые расстояния, а так же позволяют многократно использовать информацию без ущерба для носителя, при этом стоимость хранения, копирования и предоставления ее, относительно низкая. Среди общегосударственных программ и проектов в области библиотек в России сегодня четко обозначены приоритеты, направленные на создание и развитие электронного ресурса и обеспечение его общедоступности с помощью Интернета.[1]

На ранних стадиях, когда только начинали появляться первые компьютеры, электронные библиотеки строились на основе маинфрейм-ориентированной информационной технологии. Такие системы стоили очень дорого и требовали много усилий на обслуживание. Через некоторое время, когда мощность компьютерной техники резко возросла, и затраты на разработку подобного программного обеспечения сократились, библиотечные информационные системы стали приобретать все большую популярность.

Преимущество подобных систем очевидно: пользователю нет необходимости перебирать бумажный каталог в поисках нужной книги, тратя на это порой очень много времени, достаточно просто ввести название или другие данные о литературе в строку запроса, и система сама выдаст результаты. Главным недостатком в работе с такой системой является то, что необходимо на каждом компьютере, с которого планируется взаимодействие с системой, устанавливать специализированное программное обеспечение, а это дополнительные затраты на установку. К тому же не каждый пользователей сможет самостоятельно разобраться с подобной программой без посторонней помощи, что вызывает необходимость организации обучения.

Для решения этой проблемы были реализованы системы на основе WEB-технологий. Такие системы не предусматривают установку на компьютер пользователя какого-либо специализированного ПО. Все что необходимо — это наличие стандартного интернет браузера, который имеется в любой современной операционной системе, и выхода во

всемирную сеть. Идея функционирования подобной системы приведена на рисунке 1.

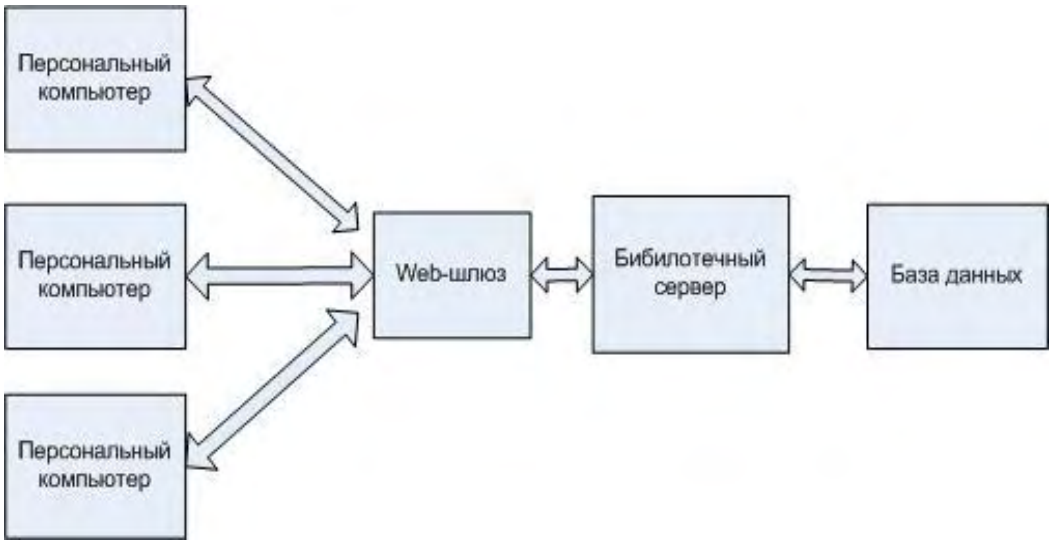


Рисунок 1 — схема передачи данных в электронной библиотеки с Web-интерфейсом.

Каждый пользователь со своего персонального компьютера посредством браузера осуществляет соединение с Web-шлюзом электронной библиотеки и передает ему данные о интересующей его литературе в формате HTML. Web-шлюз в свою очередь переводит полученные данные во внутренний формат представления сервера, после чего соединяется с библиотечным сервером и передает ему информацию о запрашиваемой литературе. Библиотечный сервер осуществляет выборку информации из базы данных и так же в своем внутреннем формате передает ее Web-шлюзу, который осуществляет перевод ее в формат HTML документа и отправляет на компьютер пользователя.

Обычному пользователю намного проще самостоятельно разобраться с Web-ориентированными приложениями, т.к. их интерфейс намного проще и понятнее. Так же несомненным плюсом в таком подходе является возможность работать с электронным каталогом с любого компьютера, подключенного к Интернет.

Потребности студентов и преподавателей вузов в получении информации в настоящее время значительно шире того, что в состоянии предоставить вузовские библиотеки. Библиотеки вузов, даже обладающие значительными собственными фондами, утрачивают роль главных информационных центров для своих читателей. Чрезвычайно сложным в настоящее время является положение с поступлением периодических изданий из стран дальнего зарубежья. Как известно, периодика, в особенности иностранная, является важнейшим источником оперативной информации. К сожалению, в данный момент эта информация практически малодоступна традиционными методами не только для значительной части студентов, но и преподавателей. Все это заставляет библиотеки изыскивать альтернативные источники комплектования. Речь идет о возможности организации в вузовских библиотеках доступа к новейшей информации из зарубежных периодических изданий с помощью телекоммуникационных сетей и организации полнотекстовых библиотек.[2]

Существует немало готовых решений для построения электронной полнотекстовой библиотеки с Web-доступом. Самыми популярными в нашей стране являются Web-ИРБИС и сервер Z39.50 с http-шлюзом. Оба этих продукта обладают следующими ключевыми особенностями:

- Единая технология обслуживания локальных и удаленных пользователей.
- Возможность работы с любым количеством библиографических баз данных.

- Возможность авторизированного обслуживания пользователей и заказа литературы
- Широкие возможности настройки и создания наиболее удобного для пользователей конкретной библиотеки окружения.
- Возможность использования форматов вывода RUSMARC, UNIMARC, USMARC.

Существенным отличием можно считать следующее: сервер Z39.50 написан на языке программирования Java, следовательно он может функционировать на любой платформе, под которую существует так называемая виртуальная машина Java, поэтому для переноса с одного типа операционной системы на другую практически не представляет сложностей. Система ИРБИС функционирует только под операционной системой Windows, следовательно для ее использования нужно покупать соответствующую ОС.

В Алтайском государственном техническом университете в настоящее время создается электронная библиотека, одним из компонентов которой будет является полнотекстовая библиотека. Для ее реализации за основу был взят сервер Z39.50 по следующим причинам:

- Поиск и извлечение информации в различных системах независимо от аппаратного и системного программного обеспечения;
- Простой доступ ко множеству удаленных баз данных при помощи одной-единственной программы;
- Обработка библиографической и любой другой информации;

Внедрение полнотекстовой библиотеки позволит:

- получать доступ только после прохождения авторизации;
- получать доступ к электронным изданиям с любого компьютера, подключенного к сети Internet;
- осуществлять поиск литературы как по ключевым словам, так и по названию, автору и т.д.;
- скачивать электронную версию литературы на локальный компьютер пользователя;
- оформлять заказ на печать литературы в типографии АлтГТУ (только для внутривузовской литературы);
- вести статистику по количеству скачанной литературы и объему переданного трафика;

Список литературы

1. Библиотеки и информационные технологии: десять лет спустя. / Режим доступа: http://lib.1september.ru/2003/21/7.htm
2. Новые информационные технологии в вузовских библиотеках России / Режим доступа: http://www.inion.ru/product/feonov13.htm
3. Введение в Z39.50 / Режим доступа: http://z3950.uiggm.nsc.ru/ansi/pubs/Z-Intro-2 /default.htm

АВТОМАТИЗАЦИЯ ПРОЦЕССА ЗАКУПОК БЮДЖЕТНОЙ ОРГАНИЗАЦИЕЙ В СООТВЕТСТВИИ С ТРЕБОВАНИЯМИ ФЕДЕРАЛЬНОГО ЗАКОНА №94 «О РАЗМЕЩЕНИИ ЗАКАЗОВ НА ПОСТАВКИ ТОВАРОВ, ВЫПОЛНЕНИЕ РАБОТ, ОКАЗАНИЕ УСЛУГ ДЛЯ ГОСУДАРСТВЕННЫХ И МУНИЦИПАЛЬНЫХ НУЖД»

Трегубова Ю.Б. – студентка, Ананьев П.И. – ст. преподаватель
Алтайский государственный технический университет (г. Барнаул)

В связи с вступлением в силу 1 января 2006 г. федерального закона № 94 "О размещении заказов на поставки товаров, выполнение работ, оказание услуг для

государственных и муниципальных нужд", который является основой нормативно-правовой базы, регулирующей отношения в данной сфере работы предприятий и организаций, остро встала потребность в сборе заявок на приобретение различных товаров и, как следствие, появилась необходимость отслеживать, что какие-то товары поступили в организацию для удовлетворения конкретной заявки, а какие-то – нет.

В бюджетной организации имеется большое число различных подразделений. Нормальное функционирование организации возможно только при синхронной бесперебойной работе всех ее подразделений. Одним из факторов, влияющих на качество работы подразделения, является наличие необходимых расходных материалов и изделий, обеспечивающих требуемые условия труда.

Для того, чтобы получить необходимые материалы, всякое структурное подразделение (отдел, подразделение) должно сформировать заявку, в дальнейшем эту заявку должны утвердить руководитель подразделения, сформировавшего заявку, отдел снабжения, отдел закупок (Рисунок 1).

В силу сложности схемы рассмотрения, утверждения и исполнения заявок, а также необходимости их быстрого исполнения, было принято решение о разработке программного продукта, позволяющего автоматизировать эти процессы. Наличие информационной системы позволит снизить вероятность случайной потери заявки, появления ошибок, связанным с ручным заполнением, уменьшить время рассмотрения, так как отпадет необходимость «физически» переносить заявку из отдела в отдел.

На данном этапе разработки определены ключевые характеристики и структурные особенности разрабатываемой информационной системы и реализованы «каркасы» некоторых ее подсистем.

Разработана реляционная база данных, для работы с которой выбрана система управления базами данных (СУБД) Oracle типа «клиент/сервер». Данная технология обладает целым рядом преимуществ по сравнению с другими технологиями, используемыми в многопользовательских СУБД.

Выбрана именно СУБД Oracle, потому что информационная система разрабатывается для АлтГТУ, и именно на эту СУБД ВУЗом куплена лицензия.

Программное обеспечение разрабатывается с использованием языка Java. Что является очевидным плюсом, так как все необходимые библиотеки, Фреймворки и т.д. обладают лицензиями, дающими право на их бесплатное использование.

В информационной системе реализован web-интерфейс. Такое решение обусловлено тем, что он позволяет взаимодействовать с различными программами через браузер, который есть на каждом компьютере, то есть уменьшаются финансовые затраты. Web-интерфейсы удобны тем, что дают возможность вести совместную работу сотрудникам, не находящимся в одном кабинете, помещении, городе.

Web-приложение написано с использованием фреймворка JavaServer Faces (JSF). JSF – это стандартный Java API для создания компонентов пользовательского интерфейса веб-приложений. JSF предоставляет в качестве инструментария готовые к использованию компоненты, которые можно быстро и легко использовать в веб-приложения. Фреймворк также используется для обработки навигация между различными страницами приложения. Он дает возможность передачи параметров между страницами.

Также, на данном этапе, воплощены в жизнь «каркас» подсистемы сбора заявок и «каркас» подсистемы складского учета, с учетом необходимости их совместной синхронной работы.

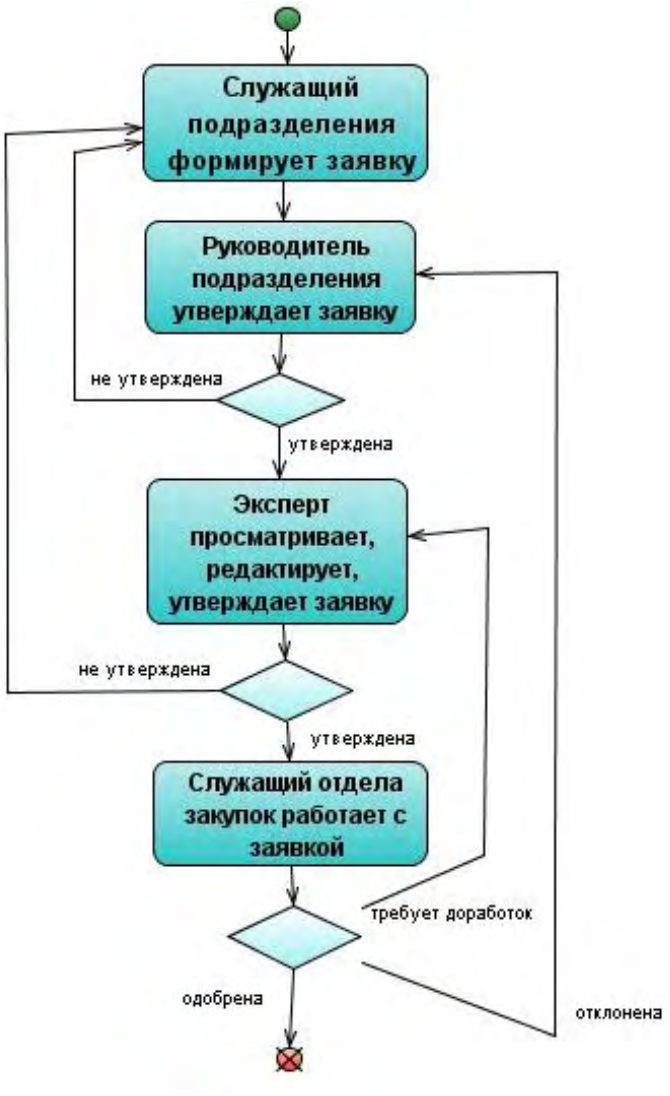


Рисунок 1 – схема рассмотрения и утверждения заявок

Система предоставляет возможность:

- формирования заявки
  - предварительная сумма по каждой строке и заявке в целом считается автоматически
- формирование внутренних приходных и расходных документов на складе с учетом имеющихся заявок
  - если на склад поступает товар для удовлетворения конкретной заявки, то этот факт сразу отмечается в приходном документе
- работы со сформированными документами (заявками, приходными и расходными документами)
  - функция редактирования даты документа недоступна, вследствие отсутствия полноценной системы авторизации и разграничения прав доступа
  - в дальнейшем, функция удаления будет доступна не всем пользователям системы

В дальнейшем будет реализовано:

- система авторизации и разграничения прав доступа. Это предоставит:
  - обеспечение защиты информации от несанкционированного доступа

    - возможность включения некоторых функций, например, корректировка даты документов
    - организацию необходимой для полноценной работы системы фильтрации заявок (пользователь видит только те заявки, с которыми он должен работать)
- дизайн пользовательского интерфейса
- проверки выполнения требуемых бизнес-правил

После внедрения данного программного решения документы, связанные с процессом сбора информации о необходимых закупках и процессом исполнения заявок и требований подразделений, будет гораздо легче формировать, они станут более точными и корректными, с точки зрения формальных требований, предъявляемых законом и руководством АлтГТУ. Также сократиться время, необходимое для формирования этих документов. Уйдет необходимость создания нескольких экземпляров одного и того же документа. Также, что немаловажно, уменьшится вероятность человеческой ошибки

Улучшения достигаются за счет использования базы данных со специфической для этих документов информацией, хранения самих документов в базе и организации автоматической передачи документов (вместо «физического» переноса документа из отдела в отдел).

Замечу, что продукт, хотя и разрабатывается для бюджетной организации, может быть с легкостью адаптирован для использования на предприятии любой формы собственности. Ведь с проблемами организации складского учета и распределения расходных материалов сталкиваются каждый день не только работники бюджетных организаций. Их приходится решать каждому предприятию: и малому, и среднему, и большому. Даже если предприятие не имеет склада в широком смысле этого слова, проблемы распределения расходных материалов его также касаются. Отличие состоит лишь в требованиях, которые предъявляет законодательство и руководство к этому процессу. А именно, к бюджетным организациям они несколько жестче, так как используются средства федерального (краевого, городского и т.д.) бюджета, хотя в коммерческих организациях требования, предъявляемые к работе склада и рассмотрению заявок, могут рассматриваться руководством еще строже.

Список литературы

1. О размещении заказов на поставки товаров, выполнение работ, оказание услуг для государственных и муниципальных нужд: Федеральный закон от 21 июля 2005 г. N 94-ФЗ // Российская газета. – 2005. – 28 июля.
2. О внесении изменений в Федеральный закон "О размещении заказов на поставки товаров, выполнение работ, оказание услуг для государственных и муниципальных нужд" и отдельные законодательные акты Российской Федерации: Федеральный закон от 30 декабря 2008 г. N 308-ФЗ // Российская газета. – 2008. – 31 декабря.

ОРГАНИЗАЦИЯ ОБМЕНА ИНФОРМАЦИЕЙ НА ОСНОВЕ БАРКОДОВ С ИСПОЛЬЗОВАНИЕМ МОБИЛЬНЫХ УСТРОЙСТВ

Третьяков И.В. – студент
Тамплон А.В. – технический директор ООО «Энтерра-Софт»
Алтайский государственный технический университет (г. Барнаул)

Сегодня компьютер, мобильный телефон и интернет проникли практически во все области повседневной жизни. С их помощью люди общаются, обмениваются фотографиями,

работают, совершают покупки, смотрят фильмы, слушают музыку. Этот список можно очень долго продолжать. Однако, все эти действия совершаются в «параллельном пространстве» работать с которым без цифровых устройств затруднительно.

Рассмотрим, например, самое простое — гиперссылки. Мы их используем, путешествуя по сети, заносим их в закладки, обмениваемся. Когда мы за компьютером, все это осуществляется без проблем — достаточно нажатия нескольких клавиш или нескольких кликов мышью. Но ситуация кардинально меняется в реальном мире. Допустим, мы увидели ссылку на интернет ресурс в журнале, газете или на рекламном щите. В этом случае уже нельзя перейти по ссылке просто нажав на нее. Ее необходимо запомнить или записать, если компьютера рядом нет. Либо точно набрать ее в адресной строке браузера, если компьютер оказался под рукой. Очевидно, что и в первом, и во втором случае возникают большие неудобства и сложности — ведь людям свойственно ошибаться.

Большинство объектов реального мира не способны поддерживать коммуникацию и отвечать на такие вопросы, как: кто, где, когда и почему, то есть, давать ту информацию, которую потребители привыкли получать, работая за компьютером.

Необходимо что-то для сведения к минимуму неудобства взаимодействия реального и цифрового мира. Именно тут и приходят на помощь 2D баркоды (иногда их так же называют физические гиперссылки).

Баркод — обычно это последовательность черных и белых полос, представляющая некоторую информацию в виде, удобном для считывания техническими средствами. Существует несколько видов:

- линейные (обычные) - называются штрихкоды, читаемые в одном направлении (по горизонтали); наиболее распространненые линейные символики: EAN (EAN-8 состоит из 8 цифр, EAN-13 — используются 13 цифр), UPC (UPC-A, UPC-E),Code39, Code128 (UCC/EAN-128), Codabar, «Interleaved 2 of 5». Линейные символики позволяют кодировать небольшой объём информации (до 20—30 символов, обычно цифр);
- двухмерными (2D баркоды) - называются символики, разработанные для кодирования большого объёма информации. Расшифровка такого кода проводится в двух измерениях (по горизонтали и по вертикали); примерами являются: PDF417, DataMatrix, QR Code и т. д. Данными баркодами можно закодировать уже около 1-2 Кб данных. [1].

Эти коды прикрепляются, наклеиваются или печатаются на предметах, что должно их сделать «умнее», дав им возможность предоставлять необходимую информацию или просто отсылать человека к соответствующей странице. Эти небольшие метки связывают людей с информацией, которая для них важна: сайты, профайлы, видео, подкасты, музыка, видео. 2D баркоды можно все чаще видеть на самых разнообразных предметах: визитных карточках, идентификационных карточках, журналах, газетах, флаерах, постерах, продуктах, головоломках, web-сайтах, CD, DVD, такси и т.д. [2]

Рассмотрим как это работает. Допустим, вы читаете журнал и видите объявление. Информация в нем вас заинтересовала, и вы захотели узнать больше. Однако, сейчас вы не можете включить компьютер и выйти для этого в интернет — возможно он физически далеко от вас или вам не хочется подходить к нему, включать, открывать браузер. А если на рекламе напечатан баркод, то вы просто достаете мобильный телефон и фотографируете его, и специальная программа распознает то, что в нем закодировано. Далее все зависит от содержимого. Допустим, это была ссылка, и, после распознавания, браузер на мобильном телефоне открывает нужную страницу.[3] Данный процесс, представленный в виде схемы,

можно видеть на рисунке 1.

Рисунок 1 – схема передачи информации посредством баркода

Но с другой стороны, например, на рекламе часто печатают электронные адреса и сейчас практически у каждого есть мобильный телефон. На первый взгляд кажется, что ничего не стоит выйти в интернет и найти всю необходимую информацию там. Но этом случае появляется другая проблема - клавиатуры мобильных устройств не удобны. Соответственно процесс набора отнимает много времени.

В итоге, очевидно, что прикрепление к реальным предметам некоторой ключевой информации добавляет им интерактивности. Но в каком виде она должна быть?

Самым распространенным на данный момент вариантом так и остается ссылка на страницу в сети интернет. Она конечно исправляет ситуацию, но так же и провоцирует проблемы: адрес в интернете сложно запомнить, высока вероятность ошибки, нужен доступ в интернет, необходимо много времени, чтобы набрать в браузере адрес, занимает много площади, желательно печатать без переносов, в большинстве случаев адрес имеет большую длину, что только ухудшает все предыдущие проблемы.

Очевидно, что эта информация должна быть представлена в том виде, который бы легко считывается электронными устройствами. Именно эта задача с успехом решается при помощи баркодов. Они обладают следующими свойствами: спроектированы специально, чтобы электронным устройствам было их легко читать, при своей небольшой площади, вмещают намного больше информации (если сравнивать с текстом), исключены ошибки чтения, можно использовать в оформлении предмета, нет необходимости запоминать, т. к. можно сразу уже узнать информацию, которая в них закодирована.

Данные которые можно закодировать в баркод не ограничиваются ссылками на интернет страницы. Популярной информацией для кодирования могут быть: данные визитной карточки, название, дата и время встречи или мероприятия, местоположение (долгота и широта), просто текст, номер телефона и многое другое.

В случае если эту информацию кодировать так, как она есть, - по сути обычным текстом - возникнет сложность в определении типа данных и их дальнейшего разбора. Чтобы это не произошло следует использовать существующие электронные форматы, которые подходят данным по смыслу. Например, для визиток существуют электронные аналоги: vCard, meCard, BIZCARD, для событий – iCalendar, vCalendar.

Необходимо также учитывать следующий момент: в некоторых случаях бывает неудобно помещать данные прямо в баркод. Тогда данные сохраняют на сервере, а в баркод заносят ссылку на эти данные. Положительным в этом подходе является: актуальность данных в баркоде (при изменении данных, не требуется изменять баркод, а соответственно и все предметы, на которые он был нанесен), для ссылки генерируется баркод меньшего размера, т. к. количество данных, которые необходимо закодировать значительно меньше.

К недостаткам относится: необходимость сервера для хранения данных, доступ к интернет устройству, которое распознает баркод.

Получение исходной информации из баркода осуществляется в несколько этапов:

1) распознавание изображение и чтение информации в баркоде;

2) анализ информации на предмет ее местоположения: если в баркоде хранилась только ссылка на изначальные данные, то скачиваем эти данные;

3) выполнение действий, которые ассоциированы с определённым типом данных;

4) выполнение дополнительных действий.

В таблице 1 приведены некоторые возможные типы данных и действие, которое необходимо выполнить при их появлении.

Таблица 1 - Типы данных и соответствующие им действия

Тип данных	Действие
Визитка	Занесение данных в адресную книгу
Событие	Добавление события в органайзер
Ссылка	Открытие встроенного браузера с данной ссылкой
СМС	Открытие редактора СМС с уже набранным номер и сообщением

Помимо стандартных действий, необходимо реализовать дальнейшее распространение полученной информации. Пользователю будет удобно, если информация с баркода добавится не только в его телефон, но и в on-line сервис, с которым он привык работать.

В результате, программный продукт, который автоматизирует все этапы (от создания баркода до реагирования на закодированную информацию), будет пользоваться популярностью. Этому способствует и возможности баркодов, и возрастающее количество людей, оценивших их преимущества, и реализация функций, которые отсутствую в аналогичных продуктах. К этим функциям относится: возможность кодирования и декодирования в различные форматы баркодов, поддержка большого числа форматов данных, работа с on-line сервисами, отсутствие ограничения на способ хранения информации в баркоде (кодировать информацию полностью или только ссылку на нее).

Список литературы

1. Barcode / Режим доступа: http://en.wikipedia.org/wiki/Barcode
2. Mainstream America is Ready for Bar Codes - Converging “Realspace” and “Mobilespace” / Режим доступа: http://harper.wirelessink.com/2006/03/29/ mainstream-america-is-ready-for-bar-codes-converging-realspace-and-mobilespace/
3. Google Introduces Physical World Hyperlinks to the U.S. / Режим доступа: http://www.centernetworks.com/google-qr-codes-print-advertising

ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ КОМПОНЕНТОВ ИНФОРМАЦИОННО – ОБРАЗОВАТЕЛЬНОЙ СРЕДЫ

Федянина О.А. – студентка, Никифоров А.Г. – к.ф.-м.н., доцент

В настоящий момент развитие информационно – коммуникационных технологий набирает все большие обороты и затрагивает почти все сферы общества. Не менее актуальным является применение данных технологий в сфере образования. Они предоставляют возможность автоматизировать процесс формирования умений, навыков и накопление необходимых с точки зрение преподавателя знаний у учащихся.

С этой целью был разработан программный комплекс «Информационно образовательная среда». Он включает в себя: приложения для оценки знаний учащихся в локальной сети, приложение для самостоятельной оценки знаний учащимися в домашних условиях, приложение администратора и образовательный портал АГТУ.

Приложения для оценки знаний учащихся в локальной сети – это приложения, предоставляющие возможность оперативно организовать и провести оценку знаний учащихся (будь то тестирование по контрольной точке, экзамен, зачет или олимпиада). Более того, оно позволяет быстро и квалифицированно оценить знания, т.е. свести к минимуму фактор человеческой ошибки. Приложения основаны на клиент – серверной технологии и все данные, необходимы для проведения мероприятия и полученные в ходе его проведения, находятся в БД на сервере.

Одно из приложений предназначено для оценки знаний группы учащихся под контролем преподавателя. Каждому учащемуся выдается свой вариант. По этому варианту ему выдается набор вопросов, дать ответы на которые он должен за ограниченное время.

Для прохождения тестирования учащемуся необходимо пройти процедуру регистрации: выбрать группу, свою фамилию и ввести пароль (номер варианта). В случае, если определенного учащегося нет в списке, он может сам себя добавить. Тем самым преподаватель освобождается от части рутинной работы по составлению списка учащихся.

Вопросы могут быть трех сложностей: А, Б и С и разбиты по разделам. В зависимости от сложности вопроса баллы, начисляемые за правильный ответ на него, ранжируются. Т.е. за правильный ответ на вопрос из части А учащийся получает 1 балл, части Б – 2, части С – 3. В соответствии с этими весами итоговый балл вычисляется по взвешенной формуле в 100-балльной системе:

$$\frac{A+2*B+3*C}{A+B+C}*100, \qquad (1)$$

где А, В, С – количество вопросов, ответы на которые даны правильно.

Вопросы могут быть 3 видов: закрытый (множественный выбор), открытый (студент сам вписывает ответ в поле) и развернутый (этот тип вопроса проверяется преподавателем).

Для иллюстрации вопроса прилагаются картинки.

Для закрытых вопросов с множественным выбором предусмотрена возможность начисления баллов, если учащийся указал не все правильные варианты ответа. В таком случае к количеству отвеченных вопросов добавляется нецелое число вида:

$$\frac{x_1 - x_2}{y}, \qquad (2)$$

где x_1 - количество отмеченных правильных вариантов, x_2 - количество отмеченных неправильных вариантов, у – всего правильных вариантов ответа. Если число получается отрицательным, то вопрос просто не засчитывается.

В процессе тестирования учащийся может пропускать вопросы и возвращаться как к пропущенным вопросам, так и к уже отвеченным. По завершении работы студенту выдается информация о том, сколько баллов он набрал по сложностям (А, Б, С) и итоговый балл.

Аналогично процесс проверки знаний проходит для другого вида приложения –

самостоятельной оценки знаний. Разница состоит лишь в том, что учащемуся не назначается вариант, а он сам выбирает разделы для проверки знаний и он не ограничен временем.

Приложение для самостоятельной оценки знаний учащимися в домашних условиях предназначено для самостоятельной оценки своих знаний с целью выявления пробелов изучении материалов. Данное приложение может быть использовано при подготовке к экзамену, олимпиаде, срезу, контрольной точки и т.п., или же просто для самостоятельного изучения предмета. Так же приложение может быть использовано для оценки знаний учащегося в удаленном варианте. В этом случае студент отсылает результаты тестирования на почту преподаватели или на образовательный портал в соответствующий раздел.

Процесс оценки знаний аналогичен описанному выше с той лишь разницей, что вопросы для тестирования загружаются из файла. Файл формируется преподавателем. Тем самым преподаватель ограждает себя от неправомерного использования вопросов учащимися, а так же регулирует уровень знаний, подбирая необходимые и полезные для изучения по его мнению вопросы и темы.

Приложение администратора предназначено для работы с данными о проведении результатов оценки знаний. для сохранения целостности системы и предотвращения несанкционированного доступа к БД, предусмотрена система аутентификации. У каждого преподавателя есть свой логин и пароль, под которыми он заходит в систему. Для оптимизации работы реализованы следующие функции.

1) Работа с данными вопросов.

Преподаватель может просматривать, добавлять, удалять и редактировать вопросы и разделы. Для более удобного восприятия информации, строится иерархия разделов (тем) и каждый вопрос относится к той или иной теме. Для каждой темы помимо списка подразделов (подтем), выводится (формируется) список вопросов. При выборе вопроса на форме отображается полная информация о нем (сложность, тип, варианты ответов с указанием правильных в случае если это закрытый вопрос, и правильный ответ если вопрос открытый, текст вопроса, список картинок с отображением выбранной). Так же как и текст, варианты ответов могут содержать не только строковый текст, но и формулы и картинки

Если к вопросу прилепляется набор картинок, то они автоматически сохраняются в так называемый архив картинок, который далее может быть так же редактируем преподавателем как список вопросов.

2) Работа с данными учащихся.

Преподаватель может работать с списками учащихся. А именно: просматривать, добавлять, удалять и редактировать данный о факультетах, группах и учащихся

3) Варианты тестов.

Вариант теста представляет собой набор (список) вопрос разной сложности по различным темам. Преподаватель может составлять вариант тестирования «вручную», т.е. добавляя в список и удаляя из него вопрос, выбранный из определенной темы. А может по шаблону.

Шаблон представляет собой список с указанием количества вопросов по сложностям. Также указывается время тестирования. Для создания вариантов тестирования по шаблону, преподаватель выбирает шаблон и указывает количества таких вариантов. По этому шаблону из списка указанных разделов случайным образом выбирается необходимое количество вопросов. Варианту присваивается пароль следующим образом: год+месяц+дата+№ (например, 200811035).

4) Статистика.

Дл просмотра результатов оценки знаний преподаватель сформировать:

- ведомость по факультету, группе или учащемуся за период или на определенную дату, в которой будет отражен список учащихся с указанием полученных баллов за части А, Б и С, выданные и итоговые баллы

- отчет о тестировании конкретного учащегося с выводом полной информации о вопросах, которые ему были выданы, ответах которые он дал и результатах тестирования

- отчет о решаемости задач по разделу, представляющий собой табличное представление и в виде диаграммы информации и количестве выдач и количестве правильных решение каждого вопроса из этого раздела.

5) Сервис

Для переноса и пополнения БД реализован импорт и экспорт данных. Преподавателем выбирается какие данные должны быть перенесены (вопросы, тестирования, шаблоны, данные о студентах и т.п.), в т.ч полностью или выборочно, после чего отобранные данные сохраняются в файл и могут из этого файл загружены заново. Тем самым организуется резервное копирование БД и возможность переноса данных, наработанных преподавателем дома.

Образовательный портал предназначен для обеспечения эффективного обучения, получения консультаций, необходимого материала не только студентов, но и школьников и абитуриентов. Поэтому, прежде всего, располагаемая на портале информация разбита и сгруппирована по целевой аудитории: школьники, абитуриенты, очное обучение, дистанционное обучение, заочное обучение

Для разграничения прав доступа предусмотрена система идентификации (а так же авторизация) пользователя. В соответствии с политикой разграничения доступа определяются возможности преподавателей, студентов и анонимного пользователя.

Так же, все данные группируются по преподавателям. Т.е. у каждого преподавателя есть своя страница или раздел, на котором он выкладывает следующую информацию:

- учебные материалы в различных форматах (Microsoft Word (.doc), Adobe Acrobat (.pdf), гипертекстовый (.htm, .html), архив с набором файлов (.zip))
- расписание занятий и консультаций с разделением (или без него) для групп учащихся
- о проведении тестирования, олимпиады с указанием параметров проведения
- проекты, приложения, программы

Помимо этого преподаватель может:

- общаться в режиме реального времени со своими коллегами и учащимися посредством почты или форума
- получать и отсылать студентам какие либо материалы
- формировать курс бучения
- проводить on-line тестирования, олимпиады
- администрировать БД тестирования в режиме on-line

Администрирование БД тестирования происходит по аналогии с программным комплексом тестирования.

Кроме того, преподаватель может организовывать так называемые on-line консультации через форум, предварительно создав тему в форуме и проинформировав студентов, что такого-то числа будет проведено обсуждение по какой-либо теме.

Помимо работы со студентами организована работа в преподавателей в группе с целью создания электронного пособия.

Для каждой из перечисленных групп организовано разграничение выдаваемой информации (информация для школьников не всего представляет интерес для студентов и наоборот).

Кроме группировки по целевой аудитории и преподавателем на сайте должно быть разделение по предметам. Аналогично разделению по учащимся, происходит разделение образовательной информации по-предметно.

Существуют два вида организации обратной связи между учащимися и преподавателями: форум и почта.

Через почту на сайте авторизованный пользователь может напрямую общаться с преподавателем:

- задать ему вопрос (проконсультироваться)
- передать работу

– сообщить необходимую информацию

Посредством форума учащийся так же может получать полезную информацию. Он может принимать участие в обсуждении какой либо темы, проблемы, решения задачи и, тем саамы, найти ответ на свой вопрос, получить консультацию у других учащихся или у преподавателя или же самому помочь другому учащемуся своим советом.

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА РАСПРЕДЕЛЕННОЙ СИСТЕМЫ УЧЕТА КАССОВЫХ ОПЕРАЦИЙ ДЛЯ СЕТИ МАГАЗИНОВ НА БАЗЕ ТЕХНОЛОГИИ J2EE

Чайка А.А. – студент, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Появление свободного рынка в начале девяностых годов двадцатого века предопределило резкий рост количества предприятий торговли различных форматов, начиная от небольших магазинов, заканчивая гипермаркетами и торговыми сетями.

Укрупнение и консолидация торговых сетей, рост конкуренции и снижение прибыльности розничного бизнеса вызывает потребность в более качественных системах управления. Главная проблема руководства компаний в данном случае – это получения актуальной и полной информации о положении дел на предприятии. И без стандартизации и автоматизации бизнес-процессов уже не обойтись. В том числе автоматизации не избежал и учет кассовых операций. Современная касса – это, по сути, компьютер с установленным программным обеспечением. Поэтому автоматизация кассового учета является неотъемлемой частью комплексной системы автоматизации.

Кроме того, активное внедрение средств автоматизации вызывает экспоненциальный рост данных, извлекать управленческую информацию, из которых становится все проблематичнее. Автоматизированные системы управления должны быть оборудованы мощными средствами бизнес - аналитики, позволяющими прогнозировать деятельность предприятия, отслеживать тенденции и выявлять закономерности. В этом случае системы мониторинга, сбора данных и их анализа выходят в современных системах управления на первый план.

Обычный набор функционала системы автоматизации торговли - учет товаров (остатки и продажи), учет контрагентов (взаиморасчеты и заказы) и финансовый учет (банк и касса). На сегодняшний день, эти задачи может решить практически любая система автоматизации приобретенная у надежного поставщика. Однако большинство таких систем не позволяет получить отчеты для производства аналитического учета, так например, для некоторых отчетов минимальной единицей учета временного интервала является один день. Очевидно, такая отчетность не может быть в дальнейшем использована в аналитических целях.

Одной из задач бизнес – аналитики выделяется задача контроля деятельности кассира и особенно его привилегированных действий. Такие данные позволяют строить отчеты по работе каждого из кассиров в разрезе различной аналитики. Особенно это задача актуально в рамках целой торговой сети, где такой контроль ручными средствами почти невозможен. Наиболее интересные отчеты позволяют посмотреть: время входа кассира в систему, операции, выполненные сторно, открытие ящика кассы, выем денег, закрытие смены. Аналитика такой отчетности в длительном периоде позволяет выявить не только факты не добросовестной работы, но и факты сговора кассиров и т.д. Такая информация позволяет дифференцированно подходить к оплате труда кассиров. Дополнительный контроль стимулирует персонал более ответственно относиться к своим обязанностям.

Решить задачу поможет распределенная система учета привилегированных кассовых операций, выполненных сотрудниками. Данная система может быть разделена на подсистемы, каждая из которых решает конкретную подзадачу.

Так автоматизированное рабочее место (АРМ) руководителя магазина позволит создавать и редактировать сотрудников конкретного магазина. Отчетность по магазинам консолидируется путем многоуровневой односторонней репликации базы данных (БД) отчетности в локальной БД каждого магазина. АРМ сотрудника отдела аналитики (безопасности) позволит анализировать собранную информацию в центральный офис (ЦО) об активности сотрудников.

Для решения задачи ускорения работы системы, придания ей мультиплатформенности и соответствия продукта современным требованиям в данной работе было решено использовать технологии на базе Java 2 Enterprise Edition (J2EE).

Таким образом, в системе можно выделить три компонента:

- АРМ руководителя магазина (Web - интерфейс);
- Многоуровневая односторонняя репликация БД;
- АРМ сотрудника отдела безопасности (просмотр отчетов по активности сотрудников) (Web -интерфейс).

Рассмотрим порядок работы системы. В магазине руководитель через web-интерфейс заводит сотрудника в локальной БД. Данные о нем с помощью репликации по цепочке поднимаются в центральный офис (ЦО). Руководитель может в любой момент заблокировать сотрудника - эти изменения также передаются в ЦО.

Каждый вечер на сервер магазина попадает отчет об активности сотрудника в течение дня (регистрации в системе, привилегированные действия и т.п.) в виде текстового файла определенного формата. Этот отчет должен загрузиться в локальную БД, а затем по репликации уйти в ЦО. В ЦО в любой момент могут через web-интерфейс посмотреть сводные отчеты по активности сотрудников по всей сети магазинов, по магазину или по сотруднику.

Рассмотрим предлагаемый подход к проектированию и разработке системы. В качестве сервера в каждом магазине было решено использовать бесплатно распространяемый сервер IBM Apache Geronimo, который содержит набор служб и сервисов, необходимых для разрабатываемой системы: контейнер web – приложений Apache Tomcat, Java Message System (JMS) брокер ActiveMQ и т.д. В качестве базы данных – Oracle Database Express Edition 10g. Для односторонней репликации БД было решено использовать средства JMS, позволяющей работать системе в асинхронном режиме. Работу с базой данных в данной работе было решено осуществлять средствами технологии Object Relation Mapping (ORM) Hibernate, позволяет работать с сущностями базы данных, как с java - объектами (Plain Old Java Object - POJO).

Для организации взаимодействия объектов системы, облегчения тестирования и разработки было решено использовать Spring framework (Spring). Spring ставит своей целью упрощение использования существующих технологий и предоставление унифицированной модели программирования - простой, но с большими возможностями. Spring – фреймворк, содержащий в себе большое число модулей для организации и управления Web-приложением: осуществления доступа к БД в связке с Hibernate, осуществления транзакций, инициализации системы обмена сообщений. Spring реализует идеологию IoC (Inversion of Control). Конфигурирование осуществляется с помощью xml файлов конфигурации, а также аннотаций в java коде. Для реализации web – интерфейса был выбран один из современных фреймворков - Apache Wicket. В основе Wicket лежит идея разделения зон ответственности кода Java и кода HTML, а также работы программистов и дизайнеров. Wicket легко интегрируется со Spring и Hibernate.

В качестве среды разработки была выбрана бесплатная платформа IDE Eclipse с модулем J2EE и набором библиотек и плагинов для web-разработки. Основным достоинством, которой является расширяемость с помощью плагинов и отличная поддержка.

Список литературы

1. RSDN – Russian Software Developer Network (российская сеть разработчиков программных продуктов): [электронный ресурс]. – Режим доступа: http://www.rsdn.ru/
2. Техническая библиотека IBM в России: [электронный ресурс]. – Режим доступа: http://www.ibm.com/developerworks/ru
3. Официальный сайт Hibernate: [электронный ресурс]. – Режим доступа: http://hibernate.org
4. Официальный сайт Spring framework: [электронный ресурс]. – Режим доступа: http://springsource.org
5. Официальный сайт Apache Wicket: [электронный ресурс]. – Режим доступа: www://wicket.apache.org
6. Сайт фирмы Auit Group Moscow - Автоматизация торговли (автоматизация магазина, сети магазинов): [электронный ресурс]. - Режим доступа: http://www.auit.ru

ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ УДАЛЁННОГО МОНИТОРИНГА И УПРАВЛЕНИЯ РАБОЧИМИ СТАНЦИЯМИ

Шашкин А.С. – студент, Боровцов Е.Г. – к.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В настоящий момент почти все, как коммерческие, так и государственные предприятия стремятся автоматизировать и оптимизировать свои бизнес-процессы. Оптимальным путем автоматизации является либо приобретение готового программного обеспечения, либо его разработка на заказ. К сожалению, приобретение соответствующего ПО является только частью решения задачи автоматизации. Следующий шаг – это интеграция нового программного продукта в существующие условия. Данная задача не всегда является тривиальной, так как предприятие может обладать обширным количеством рабочих мест. Соответственно данная задача ложится на плечи сетевых администраторов.

С другой стороны, существуют предприятия с уже налаженной инфраструктурой, но даже в этом случае количество должностных обязанностей у системных администраторов и время, которое они тратят на поддержание работоспособности инфраструктуры, не уменьшается. В данном случае перед администраторами стоят задачи:

- поддержание актуальности версий установленных продуктов,
- своевременная установка обновлений,
- специфическая настройка групп, либо определенных рабочих станций.

В связи с актуальностью перечисленных выше проблем был разработан программный комплекс, позволяющий централизованно управлять обширным количеством рабочих станций. Данный комплекс имеет модульную структуру, что позволяет устанавливать разные части системы на различные сервера. Так же при разработке была учтена текущая экономическая обстановка, что привело к поддержке большого числа СУБД (Ms SQL, MySQL, Oracle, Firebird).

В целом архитектура разработанного комплекса позволяет провести его внедрение без увеличения существующих серверных мощностей или покупки дополнительного программного обеспечения.

В ходе исследований было принято решение о реализации программного комплекса на основе клиент-серверной архитектуре.

В целом комплекс состоит из следующих модулей:

- веб-сервер,
- серверное ядро комплекса,
- клиент для рабочей станции,
- клиент администратора,

– СУБД.

Исходя из набора модулей в системе видно, что доступ к ней можно получить как через веб-интерфейс, так и из «толстого» клиента.

Одной из центральных частей комплекса является система управления базами данных (СУБД). Существует возможность использования как одной СУБД, так и нескольких, например, для изоляции данных журнала аудита.

В итоге в окончательную версию программного комплекса был включен следующий функционал:

1) консолидация информации о состоянии ПК;
2) консолидация информации об аппаратных комплектующих;
3) консолидация информации об установленном программном обеспечении;
4) возможность установки обновлений для операционной системы;
5) возможность создания установочных пакетов на основе действий пользователя;
6) возможность создания установочных пакетов на основе изменений в системе;
7) возможность устанавливания нового программного обеспечения;
8) возможность выполнения пользовательских shell-скриптов и PowerShell-скриптов на удаленных ПК;
9) возможность удаленного управления процессами на ПК;
10) возможность получения и хранения снимков экрана удаленных ПК;
11) сохранение истории обо всех действиях пользователей комплекса;
12) разграниченный доступ к возможностям комплекса.

Данный функционал позволит в кратчайшие сроки устанавливать новое программное обеспечение и более гибкое управлять удаленными рабочими станциями. По приблизительной оценке, внедрение данного комплекса в существующую инфраструктуру компьютерных классов кафедры Прикладной Математики позволит экономить до 60% времени обслуживающего персонала, тем самым произойдет повышение качества обслуживания компьютерной техники и скорость реагирования на неисправности.

При разработке были использованы такие технологии как Windows .Net Framework, ADO.net Entity Framework, библиотека для C++ MFC. Для написания программного комплекса использовалась среда Microsoft Visual Studio 2008. В качестве первичного СУБД сервера использовался Ms SQL Server и Firebird SQL Server.

МОДУЛЬ ОБЩЕЙ СТАТИСТИКИ И СТАТИСТИКИ ВОСТРЕБОВАННОСТИ УЧЕБНЫХ МАТЕРИАЛОВ ДЛЯ ЭЛЕКТРОННОЙ БИБЛИОТЕКИ АЛТГТУ

Якушев А.Е. – студент, Андреева А.Ю. – к.ф.м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Оценка предпочтений пользователей играет огромную роль в работе современных Web-сайтов сети интернет. Наличие множества одноплановых (посвящённых сходной тематике) ресурсов в сети требует усиления борьбы за каждого посетителя. А это подразумевает под собой не только попытки привлечь пользователя на сайт, но и удержать его, предложив (помимо более актуальной и подробной информации) более удобный интерфейс и систему функционирования (использования возможностей ресурса).

Преследуя описанные выше цели, владельцы Web-сайтов активно используют различные модули статистики (зачастую сразу по нескольку экземпляров от различных разработчиков). Данные системы позволяют оценить предпочтения посетителей: какие страницы более популярны, с каких наблюдается частый переход на другие сайты (обрывается последовательность переходов), какую информацию скачивают с сайта. Более того, с помощью модулей статистики становится возможным оценить (и, возможно, сравнить с аналогами – при наличии подобной информации в открытом доступе) посещаемость сайта

– общую и отдельно для уникальных («новых») пользователей. В результате можно делать выводы о популярности и целесообразности активной поддержки ресурса.

Все существующие системы сбора статистики реализованы в одном из двух вариантов: счётчики и лог-анализаторы [1]. Счётчики просты и удобны в использовании, позволяют получать оперативную информацию, а главное – они наглядны. С другой стороны, подобные реализации не лишены недостатков. Во-первых, для установки счётчика обязательно внедрение на сайт программного кода. Во-вторых, возможен факт утери или неполного сбора данных в случае, если Web-страница не до конца загрузилась или возникли технические сбои передачи информации. Наконец, существование сложностей со сбором специфической информации. Данные по трафику сайта (объёму или потоку информации, передаваемой по сети), а также закладкам, поставленным пользователями на сайт в своём браузере получить не возможно. Для сбора информации по скачиваемому контенту (содержимому) приходится вносить изменения в настройках отображения и работы сайта.

Лог-анализаторы лишены недостатков счётчиков. Они помогают получить более точные данные о числе посетителей сайта, а также решать более узкие и сложные задачи, создавать собственные специфические отчёты, анализировать ошибки работы сервера и отслеживать хакерские атаки. Однако подобные реализации обладают одним серьёзным недостатком, сводящим на нет возможность широкого использования лог-анализаторов – необходимость достаточно высокой квалификации пользователей программы.

Современные реализации счётчиков можно условно разделить на два больших класса. В первый попадают модули статистики, использующие однозначно определённую разработчиком (располагающуюся на специализированном сервере). Помимо очевидного плюса – отсутствия необходимости заботиться о подготовке базы данных и её функционировании, данный способ имеет большие минусы. Во-первых, данные, собранные подобными счётчиками становятся доступны не только владельцам сайта, но и разработчикам модуля и, как правило, используются в различных маркетинговых исследованиях. Этот фактор сводит на нет возможности применения подобных счётчиков на конфиденциальных страницах ресурса, являясь очевидной утечкой косвенной информации. Во-вторых, работа счётчика теперь зависит от работы сайта производителя (разработчика). Однако, несмотря на вышесказанное, данные модули чрезвычайно популярны, что связано в основном с их бесплатностью и простотой настройки.

Во второй класс модулей статистики попадают системы, использующие собственные базы данных, размещаемые на серверах, определённых пользователем сервиса. Подобные системы менее популярны, так как почти все они являются платными. Однако, если цель использования модуля – именно сбор определённых (и очень часто контекстно-зависимых данных), то данные счётчики являются предпочтительным вариантом. Тем более что разработчики предоставляют систему на условиях двустороннего договора, а, следовательно, появляется возможность потребовать определённых гарантий работы.

Кроме описанных отличий двух классов модулей статистики, стоит отметить и качественное отличие предоставляемого функционала. Первый класс обычно обладает огромными возможностями анализа, которые, как правило, используются не более чем наполовину. Они разработаны для широкой аудитории и стремятся максимально удовлетворить стандартные потребности сбора статистики для некоторого («среднестатистического») ресурса. Второй класс представляют более частные счётчики, имеющие меньший функционал. Однако возможности данных систем используются почти на 100%, так как сама система модифицируется перед установкой в зависимости от потребностей конечного пользователя.

Рассмотрев работу модулей статистики и их базовые возможности, было принято решение реализовать собственный модуль статистики в виде счётчика, использующего базу данных, определённую пользователем. Впоследствии система была реализована.

При создании модуля была спроектирована база данных, предназначенная для хранения собранной информации. Сам модуль был создан с использованием следующих технологий:

- язык программирования PHP – интерфейс настройки параметров сбора статистики и просмотра его результатов;
- СУБД mySQL – хранение собранных данных и настроек модуля;
- язык программирования JavaScript и технология Ajax – клиентская часть (для размещения и функционирования счётчика на сайте пользователя);
- формат обмена данных JSON – для организации обмена данными между сервером и клиентом (Ajax).;
- PHP framework Symfony – реализация основных взаимодействий всех используемых технологий, а также упрощение их использования.
- JavaScript framework jQuery – реализация основного интерактивного функционала страниц (на стороне клиента, с использованием JavaScript).

В итоге был разработан модуль со следующими базовыми возможностями:

- Работа с собственной (локальной или удалённой) базой данных.
- Сбор общей статистики посещений пользователей.
- Сбор статистики скачивания файлов, размещённых на страницах ресурса.
- Сбор подробной статистики для отдельных страниц ресурса.
- Отображение географии пользователей.
- Интеграция данных о правах пользователей с существующей базой данных электронной библиотеки АлтГТУ.
- Отслеживание последовательностей переходов пользователей (по страницам ресурса).
- Отображение панелей настройки работы счётчика на страницах целевого сайта.
- Отображение графиков посещаемости.

Список литературы

1. Википедия, свободная энциклопедия [электронный ресурс] – Режим доступа: http://ru.wikipedia.org/
2. PHP framework Symfony [электронный ресурс] – Режим доступа: http://www.symfony-project.org/
3. JavaScript framework jQuery [электронный ресурс] – Режим доступа: http://jquery.com/
4. Описание формата JSON [электронный ресурс] – Режим доступа: http://www.json.org/

ПРОБЛЕМА РАЗРАБОТКИ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ДЛЯ КАЗУАЛЬНЫХ ИГРОВЫХ ПРОГРАММ

Дрыганец Н.С, Мамонтова Е.В. – студентки, Крючкова Е.Н. – к.ф.-м.н., профессор

Графический Интерфейс Пользователя (GUI) - система средств для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов (окон, кнопок, меню, списков и т. п.). При этом в отличие от интерфейса командной строки, пользователь имеет произвольный доступ (с помощью клавиатуры или устройства координатного ввода - вроде мыши) ко всем видимым экранным объектам. Впервые концепция GUI была предложена учеными из исследовательской лаборатории Xerox PARC в 1970-х, но получила коммерческое воплощение лишь в продуктах корпорации Apple Computer. В настоящее время GUI является стандартной составляющей большинства доступных на рынке операционных систем и приложений.

Задача GUI - максимально упростить и визуализировать взаимодействие пользователя и приложения. На сегодняшний существует множество похожих по своему функциональному назначению приложений и зачастую основным отличием для рядового пользователя

становится именно интерфейс. Целые команды разработчиков, психологов и других специалистов трудятся, чтобы найти какую-то общую концепцию, по которой бы графический интерфейс считался бы удобным для всех.

В этой сфере выделяется игровая индустрия, в ней необходимы какие-то особые правила, потому что игра - это моделирование окружающей действительности. Играющий человек мысленно погружается в другой мир, где можно ходить, бегать, прыгать, стрелять, говорить... Однако, несмотря на все визуальные отличия, на уровне машины все содержимое экрана остается тем же самым: кнопки, курсор, картинки, окна...

Рассмотрим проблемы создания интерфейса для казуальных игр, которые характеризуются упрощенным управлением, осуществляемым с помощью манипулятора типа мышь, игрок видит все сцены лишь под одним углом, целевая аудитория казуальных игр – непродвинутые пользователи, которые смогли бы понять, как осуществляется управление за пару минут.

На сегодняшний день существует массы стандартных средств разработки, и на первый взгляд очевидным кажется использование именно этих средств. Но на деле для интерфейса игры они малопригодны ввиду ограниченности их возможностей. Каждая игра должна быть яркой и индивидуальной, иметь свою неповторимую стилистику.

Достичь этого при помощи стандартных средств представляется маловероятным. К тому же, основная задача разработчика - увеличить производительность игры, для чего нужно как-то оптимизировать размеры графических изображений и способы их подгрузки в игру. А если переводить игры на десять языков? Придется переделывать весь дизайн под каждую версию игры? Очевидно, что это слишком громоздкое решение, поэтому нужна какая-либо альтернатива.

На самом деле далеко за примерами другой реализации графического интерфейса ходить не надо. Достаточно лишь посмотреть на web-страницы. Язык разметки позволяет реализовать очень гибкий, функциональный, индивидуальный и красочный интерфейс: для кардинального изменения внешнего вида страницы достаточно поменять (добавить, если его не было) файл с таблицей стилей.

Ну, и так как HTML (Hyper Text Markup Language) – это лишь частный случай разметочного языка, рассматриваем XML (eXtensible Markup Language).

XML (расширяемый язык разметки) — рекомендованный Консорциумом Всемирной паутины язык разметки, фактически представляющий собой свод общих синтаксических правил. XML — текстовый формат, предназначенный для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами. Целью создания XML было обеспечение совместимости при передаче структурированных данных между разными системами обработки информации, особенно при передаче таких данных через Интернет.

Итак, использование XML позволяет:

- Упростить создание необычного интерфейса, сделать его более гибким и легко изменяемым
- Отделить программную реализацию игры от ее оформления

Однако есть также и минусы:

- Размер XML документа существенно больше бинарного представления тех же данных. В грубых оценках величину этого фактора принимают за 1 порядок (в 10 раз).
- XML содержит мета-данные (об именах полей, классов, вложенности структур), и одновременно XML позиционируется как язык взаимодействия открытых систем. При передаче между системами большого количества объектов одного типа (одной структуры), передавать метаданные повторно нет смысла, хотя они содержатся в каждом экземпляре XML описания.
- Неоднозначность моделирования. Нет общепринятой методологии для моделирования данных в XML, в то время как для реляционной модели и объектно-ориентированной

такие средства разработаны и базируются на реляционной алгебре, системном подходе и системном анализе.

В связи с избыточностью информации, создание документов XML вручную – очень трудоёмкая работа, поэтому предлагается разработка визуального редактора GUI, которая бы позволила по сформированному внешнему виду документа создать структуру xml-файла, соответствующего данному документу.

Предлагается внедрение разработки в уже существующую систему GUI, которая основана на следующих принципах:

- Вся информация внутри документа расположена в контейнерах, имеющих определенную ориентацию: горизонтальную и вертикальную. Это означает, что все компоненты, расположенные в контейнере, вставляются последовательно либо вертикально, либо горизонтально. Компонент контейнера может быть как очередным контейнером, так и конечным функциональным элементом (кнопка, надпись, картинка и т.д.). Для более сложной настройки есть возможность указать абсолютные координаты для компонента, но при малейших изменениях внешнего вида есть большая вероятность, что этот компонент будет отображаться неправильно.
- Базовым контейнером документа является окно.
- Для каждого компонента указывается один из существующих шаблонов внешнего вида для компонентов этого типа, идентификатор компонента, номер слоя, на котором отображается компонент, название группы скриптов, которые выполняют обработку событий.

РАЗРАБОТКА ПОРТАЛОВ НА ОСНОВЕ ТЕХНОЛОГИИ JSF

Дрыганец Н.С, Мамонтова Е.В. – студентки, Крючкова Е.Н. – к.ф.-м.н., профессор

Разработка порталов является сегодня актуальной задачей, так как порталы получили широкое распространение в сети интернет и используются как для развлекательных, так и для образовательных и рекламных целей.

Функционирование портала подразумевает разграничение прав доступа пользователей к различным разделам и возможностям, которые может предоставить система.

Рассмотрим проблемы разработки систем регистрации и аутентификации пользователей портала. Принципиальным моментом при разработке таких систем является обеспечение существования групп пользователей, реализация разграничения прав доступа пользователей к различным разделам в зависимости от того, к какой группе они относятся, разграничение получаемого содержимого web-страниц. При разработке системы нами была выбрана технология Java Server Faces (JSF). Выбор технологии был обусловлен следующим:

- JSF является бесплатным фреймворком с довольно обширными возможностями, а значит, он более привлекателен в глазах разработчиков и пользователей чем, к примеру, ASP.Net
- JSF, как и все, связанное с языком Java, является кроссплатформенным
- Данный фреймворк позволяет использовать все возможности языка Java при обработке содержимого web-страниц.
- Java Server Faces включает в себя набор API для отображения UI компонентов, управления их состоянием, отслеживания событий, проверки пользовательского ввода, определения навигации между страницами и поддержки интернационализации и библиотеку тегов JSP для отображения JSF интерфейса посредством JSP-страниц. Таким образом, с помощью этого набора тегов можно, к примеру, в несколько строк

без циклов отобразить на странице в табличном виде результат запроса к БД. Надо сказать, что PHP – другой бесплатный кроссплатформенный язык, подобным похвастаться не может – при формировании страниц неизбежно появляются циклы. И, т.к. он использует компоненты JavaScript, периодически извлечение данных, введенных пользователем на странице, получается не столь прозрачным, как при использовании JSF.

Учитывая особенности JSF, наше приложение реализовано как взаимосвязанная структура из следующих трех компонент:

- Набор Java-классов (в данной технологии их называют Java-bean), реализующих общую логику приложения.
- Набор JSP-страниц, содержащих непосредственно информацию, которая будет видна пользователю. Компоненты страниц могут использовать информацию и методы из Java-bean'ов.
- Конфигурационные xml-файлы faces-config.xml и web.xml. В файле web.xml хранится информация обо всех используемых в приложении сервлетах, о стартовой странице приложения, о настройках сессий. В файле faces-config.xml содержится информация о доступных для JSP-страниц Java-классах приложения, навигационные правила перемещения между jsp-страницами при указанных действиях.

Таким образом, над приложением легко могут работать параллельно дизайнеры, отвечающие за внешний вид сайта, которым программисты указывают, какие свойства и методы они могут использовать для каких-то конкретных задач, и программисты, описывающие ядро приложения. Такой подход позволяет инкапсулировать данные, которые нельзя вызывать с web-страниц напрямую. К примеру, класс, отвечающий за соединение к базе данных и выполнение запросов к ней, должны использовать только другие классы, а не непосредственно пользователь, иначе могут возникнуть ситуации с незавершенными транзакциями, незакрытыми соединениями и порчей данных из базы.

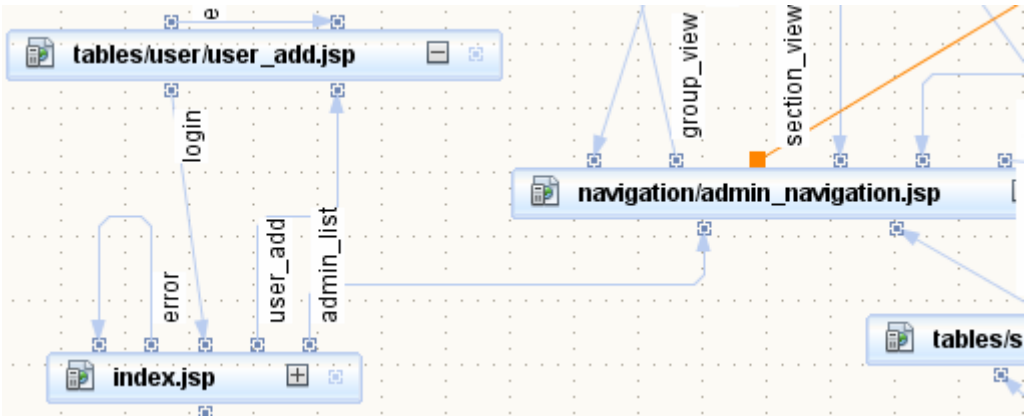
Спроектированное web-приложение приобретает четкую модульную структуру, в нем можно применять различные паттерны проектирования, что позволяет приблизить его к десктопному приложению.

Интересной особенностью является система навигационных правил между jsp-страницами. У каждого JSF-компонента есть свойство action, которое задается текстовой строкой. Это свойство означает переход между двумя jsp-страницами. Куда производится переход со страницы, прописывается в файле faces-config.xml. В текстовом виде выглядит это так:

```
<navigation-rule>
    <from-view-id>/tables/user/user_view.jsp</from-view-id>
    <navigation-case>
      <from-outcome>user_edit</from-outcome>
      <to-view-id>/tables/user/user_edit.jsp</to-view-id>
    </navigation-case>
    <navigation-case>
      <from-outcome>admin_navigation</from-outcome>
      <to-view-id>/navigation/admin_navigation.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
```

Среда разработки по структуре файла faces-config.xml создает визуальное представление в виде диаграммы связанных web-страниц, где связь соответствует параметрам в xml-файле.

Это выглядит так:



В принципе, это позволяет упростить для разработчиков понимание о том, какое перемещение подразумевается под этим действием: при ошибке переместились туда, при регистрации переместились на форму регистрации, при авторизации переместились на стартовую страницу пользователя и т.д. Не надо работать с далеко не всегда понятными URL, включающими в себя ключи в базах данных и мало ли чего еще. Также если вдруг при возникновении ошибки понадобилось перемещаться на другую страницу, не надо искать все старые перемещения и исправлять там ссылку – достаточно лишь по-другому протянуть стрелку в xml-файле. Но, как это часто бывает, здесь есть и свои минусы. При сложной структуре jsp-страниц вполне возможна и картина, перегруженная множеством связей.